PP/018/3/3rd Revision

LABORATORIUM
INFORMATICS ENGINEERING
FACULTY OF INDUSTRIAL TECHNOLOGY
AHMAD DAHLAN UNIVERSITY

PRACTICUM MODULE

DATABASE

(1835341)

Team :
Murein Miksa Mardhia, S.T., M.T.
Bayu Alif Alfarisqi

2020

HAK CIPTA

PETUNJUK PRAKTIKUM BASIS DATA

Diterbitkan oleh:
Program Studi Teknik Informatika
Fakultas Teknologi Industri
Universitas Ahmad Dahlan
Jalan Ring Road Selatan, Tamanan, Banguntapan, Bantul Yogyakarta 55166

Penulis	: Murein Miksa Mardhia, S.T., M.T. Bayu Alif Alfarisqi
Editor	: Laboratorium Teknik Informatika, Universitas Ahmad Dahlan
Desain sampul	: Laboratorium Teknik Informatika, Universitas Ahmad Dahlan
Tata letak	: Laboratorium Teknik Informatika, Universitas Ahmad Dahlan
Ukuran/Halaman	: 21 x 29,7 cm / 76 halaman

Didistribusikan oleh:

Laboratorium Teknik Informatika
Universitas Ahmad Dahlan
Jalan Ring Road Selatan, Tamanan, Banguntapan, Bantul Yogyakarta 55166
Indonesia

PREFACE

Our gratitude goes to the presence of Allah SWT who has bestowed His grace and guidance so that we can complete the PRACTICUM GUIDE MODULE for DATABASE COURSE for The International Class of Academic Year 2020/2021 in the Informatics Engineering Department, Ahmad Dahlan University.

The material presented in this practicum guidance module has been adapted to Database syllabus planning, main reference of (Silberschatz, Korth, & Sudarshan, 2011), used both in regular and international classes. Each meeting contains an explanation of the theory related to the material provided and an explanation of the stages of work that must be carried out while practicing in the laboratory.

We realize that there are still many imperfections in this writing. We always accept criticism and suggestions for improving the quality of guidelines and practicum implementation.

We thank the management team of the International Class of Informatics Engineering Department along with practicum assistant Mr. Bayu Alif Alfarisqi who was involved in making this practicum guide. Hopefully the results obtained from the implementation of the Database practicum through this practicum guide can provide benefits and contributions in the advancement of science.

Yogyakarta, September 2020

Team

Authors

Murein Miksa Mardhia, S.T., M.T.

Bayu Alif Alfarisqi

REVISION HISTORY

The undersigned below:

Name : Murein Miksa Mardhia, S.T., M.T.

NIK/NIY : 60160960

Position : Database Lecturer for International Class

Hereby declare that the Revision of the Database Practicum Guidelines for the Informatics Engineering Department of Ahmad Dahlan University has been carried out with the following explanation:

No	Revision	Date	Doc Number
1	a. Upgrading Tool from Ms Access to XAMPP	15 Agustus 2018	
2	Reformatting to the newest template + English Translation	7 September 2019	
3	Added Indexing material	24 september 2020	

Yogyakarta, September 25th, 2020

Sincerely,

Murein Miksa Mardhia, S.T., M.T

NIK/NIY. 60160960

LETTER OF STATEMENT

The undersigned below:

Name : Murein Miksa Mardhia, S.T., M.T.

NIK/NIY : 60160960

Position : Database Lecturer of International Class

Explain clearly that this Practicum Module has been reviewed and will be used for practicum activity in Academic Year 2019-2020 in Multimedia Laboratory, Informatics Engineering Department, Faculty of Industrial Technology, Ahmad Dahlan University.

Yogyakarta, 7 September 2020

Mengetahui,

Ketua Kelompok Keilmuan Rekayasa Perangkat Lunak dan Data (RELATA)	Kepala Laboratorium Praktikum Teknik Informatika
	
Drs., Tedy Setiadi, M.T.	**Lisna Zahrotun, S.T., M.Cs.**
NIY. 60030475	NIY. 60150773

VISI DAN MISI PRODI TEKNIK INFORMATIKA

VISI

Menjadi Program Studi Informatika yang diakui secara internasional dan unggul dalam bidang Informatika serta berbasis nilai-nilai Islam.

MISI

1. Menjalankan pendidikan sesuai dengan kompetensi bidang Informatika yang diakui nasional dan internasional
2. Meningkatkan penelitian dosen dan mahasiswa dalam bidang Informatika yang kreatif, inovatif dan tepat guna.
3. Meningkatkan kuantitas dan kualitas publikasi ilmiah tingkat nasional dan internasional
4. Melaksanakan dan meningkatkan kegiatan pengabdian masyarakat oleh dosen dan mahasiswa dalam bidang Informatika.
5. Menyelenggarakan aktivitas yang mendukung pengembangan program studi dengan melibatkan dosen dan mahasiswa.
6. Menyelenggarakan kerja sama dengan lembaga tingkat nasional dan internasional.
7. Menciptakan kehidupan Islami di lingkungan program studi.

LABORATORIUM RULES AND REGULATIONS

DOSEN/KOORDINATOR PRAKTIKUM

1. Dosen harus hadir saat praktikum minimal 15 menit di awal kegiatan praktikum dan menandatangani presensi kehadiran praktikum.
2. Dosen membuat modul praktikum, soal seleksi asisten, pre-test, post-test, dan responsi dengan berkoordinasi dengan asisten dan pengampu mata praktikum.
3. Dosen berkoordinasi dengan koordinator asisten praktikum untuk evaluasi praktikum setiap minggu.
4. Dosen menandatangani surat kontrak asisten praktikum dan koordinator asisten praktikum.
5. Dosen yang tidak hadir pada slot praktikum tertentu tanpa pemberitahuan selama 2 minggu berturut-turut mendapat teguran dari Kepala Laboratorium, apabila masih berlanjut 2 minggu berikutnya maka Kepala Laboratorium berhak mengganti koordinator praktikum pada slot tersebut.

PRAKTIKAN

1. Praktikan harus hadir 15 menit sebelum kegiatan praktikum dimulai, dan dispensasi terlambat 15 menit dengan alasan yang jelas (kecuali asisten menentukan lain dan patokan jam adalah jam yang ada di Laboratorium, terlambat lebih dari 15 menit tidak boleh masuk praktikum & dianggap Inhal).
2. Praktikan yang tidak mengikuti praktikum dengan alasan apapun, wajib mengikuti INHAL, maksimal 4 kali praktikum dan jika lebih dari 4 kali maka praktikum dianggap GAGAL.
3. Praktikan harus berpakaian rapi sesuai dengan ketentuan Universitas, sebagai berikut:
   a. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki-laki / Perempuan) dan Topi harus Dilepas.
   b. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
   c. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki-laki / Perempuan).
   d. Laki-laki tidak boleh memakai gelang, anting-anting ataupun aksesoris Perempuan.
4. Praktikan tidak boleh makan dan minum selama kegiatan praktikum berlangsung, harus menjaga kebersihan, keamanan dan ketertiban selama mengikuti kegiatan praktikum atau selama berada di dalam laboratorium (tidak boleh membuang sampah sembarangan baik kertas, potongan kertas, bungkus permen baik di lantai karpet maupun di dalam ruang CPU).
5. Praktikan dilarang meninggalkan kegiatan praktikum tanpa seizin Asisten atau Laboran.
6. Praktikan harus meletakkan sepatu dan tas pada rak/loker yang telah disediakan.
7. Selama praktikum dilarang NGENET/NGE-GAME, kecuali mata praktikum yang membutuhkan atau menggunakan fasilitas Internet.
8. Praktikan dilarang melepas kabel jaringan atau kabel power praktikum tanpa sepengetahuan laboran
9. Praktikan harus memiliki FILE Petunjuk praktikum dan digunakan pada saat praktikum dan harus siap sebelum praktikum berlangsung.
10. Praktikan dilarang melakukan kecurangan seperti mencontek atau menyalin pekerjaan praktikan yang lain saat praktikum berlangsung atau post-test yang menjadi tugas praktikum.
11. Praktikan dilarang mengubah setting software/hardware komputer baik menambah atau mengurangi tanpa permintaan asisten atau laboran dan melakukan sesuatu yang dapat merugikan laboratorium atau praktikum lain.

12. Asisten, Koordinator Praktikum, Kepala laboratorium dan Laboran mempunyai hak untuk menegur, memperingatkan bahkan meminta praktikan keluar ruang praktikum apabila dirasa anda mengganggu praktikan lain atau tidak melaksanakan kegiatan praktikum sebagaimana mestinya dan atau tidak mematuhi aturan lab yang berlaku.
13. Pelanggaran terhadap salah satu atau lebih dari aturan diatas maka Nilai praktikum pada pertemuan tersebut dianggap 0 (NOL) dengan status INHAL.

ASISTEN PRAKTIKUM

1) Asisten harus hadir 15 Menit sebelum praktikum dimulai (konfirmasi ke koordinator bila mengalami keterlambatan atau berhalangan hadir).
2) Asisten yang tidak bisa hadir WAJIB mencari pengganti, dan melaporkan kepada Koordinator Asisten.
3) Asisten harus berpakaian rapi sesuai dengan ketentuan Universitas, sebagai berikut:
   a. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki-laki / Perempuan) dan Topi harus Dilepas.
   b. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
   c. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki-laki / Perempuan).
   d. Laki-laki tidak boleh memakai gelang, anting-anting ataupun aksesoris Perempuan.
4) Asisten harus menjaga kebersihan, keamanan dan ketertiban selama mengikuti kegiatan praktikum atau selama berada di laboratorium, menegur atau mengingatkan jika ada praktikan yang tidak dapat menjaga kebersihan, ketertiban atau kesopanan.
5) Asisten harus dapat merapikan dan mengamankan presensi praktikum, Kartu Nilai serta tertib dalam memasukan/Input nilai secara Online/Offline.
6) Asisten harus dapat bertindak secara profesional sebagai seorang asisten praktikum dan dapat menjadi teladan bagi praktikan.
7) Asisten harus dapat memberikan penjelasan/pemahaman yang dibutuhkan oleh praktikan berkenaan dengan materi praktikum yang diasisteni sehingga praktikan dapat melaksanakan dan mengerjakan tugas praktikum dengan baik dan jelas.
8) Asisten tidak diperkenankan mengobrol sendiri apalagi sampai membuat gaduh.
9) Asisten dimohon mengkoordinasikan untuk meminta praktikan agar mematikan komputer untuk jadwal terakhir dan sudah dilakukan penilaian terhadap hasil kerja praktikan.
10) Asisten wajib untuk mematikan LCD Projector dan komputer asisten/praktikan apabila tidak digunakan.
11) Asisten tidak diperkenankan menggunakan akses internet selain untuk kegiatan praktikum, seperti Youtube/Game/Medsos/Streaming Film di komputer praktikan.

LAIN-LAIN

1. Pada Saat Responsi Harus menggunakan Baju Kemeja untuk Laki-laki dan Perempuan untuk Praktikan dan Asisten.
2. Ketidakhadiran praktikum dengan alasan apapun dianggap INHAL.
3. Izin praktikum mengikuti aturan izin SIMERU/KULIAH.
4. Yang tidak berkepentingan dengan praktikum dilarang mengganggu praktikan atau membuat keributan/kegaduhan.
5. Penggunaan lab diluar jam praktikum maksimal sampai pukul 21.00 dengan menunjukkan surat ijin dari Kepala Laboratorium Prodi Teknik Informatika.

Yogyakarta, 15 Februari 2019

Kepala Laboratorium Jaringan

Lisna Zahrotun, S.T., M.Cs

NIK/NIY. 60150773

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE LIST

MEETING 1: UNDERSTANDING THE CASE WITH ENTITY IDENTIFICATION AND DATA PROCESSING

Meeting : 1

Time Allocation : 150 minutes
- Pre-Test : 30 minutes
- Prakticum : 90 minutes
- Post-Test : 30 minutes

Scoring : 100%
- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

1.1. GOALS AND PERFORMANCE INDICATORS

By following this session, students are expected:

1. Able to get to know MySQL and Database Management System.
2. Able to understand table, data type and primary key within Database Management System.
3. Able to identify entity from case(s) provided.
4. Able to process data in a table which includes adding data, deleting data and editing data

Performance indicators are measured by:
1. Create Database
2. Create Table
3. Ability to do Insert, Update, Delete

1.2. STUDY LITERATURE

1.2.1. Tables and Attributes

Entities or tables are individuals or objects that have something tangible (existence) and can be distinguished from something else. Attributes are characteristics that define entities so that they can distinguish between one entity and another. An entity must have one or more attributes to define the characteristics of that entity. Examples of attributes from Student Entities are name, name, class.

1.2.2. Get to Know MySQL

MySQL is a database server program that capable of receiving and sending data very quickly, multi-user and using the basic command SQL (Structured Query Language). MySQL is two forms of licensing, namely Free Software and Shareware. MySQL that we usually use is MySQL Free Software which is under the GNU / GPL (General Public License) License. MySQL is a free

database server, meaning we are free to use this database for personal or business needs without having to buy or pay for licenses.

MySQL was first pioneered by a database programmer named Michael Widenius. In addition to the database server, MySQL is also a program that can access a MySQL database that is positioned as a Server, which means our program is positioned as a Client. Hence, MySQL is a database that can be used as a client or server. MySQL database is a database software in the form of a relational database or called the Relational Database Management System (RDBMS) which uses a request language called SQL (Structured Query Language).

The MySQL database has several advantages over other databases, including:

a. MySQL is a Database Management System (DBMS).
b. MySQL as a Relation Database Management System (RDBMS) or referred to as a Relational database.
c. MySQL Is a free database server, meaning that we are free to use this database for personal or business needs without having to buy or pay for licenses.
d. MySQL is a database client.
e. MySQL able to accept queries that are capped in a single request or Multi Threading.
f. MySQL is a database that able to store data with very large capacity up to GigaByte size though.
g. MySQL is supported by the ODBC driver, meaning that the MySQL database can be accessed using any application, including visuals such as Visual Basic, PHP and Delphi.
h. MySQL is a multi-user Database Server, meaning that this database is not only used by one person but can be used by many users.
i. MySQL supports fields that are used as primary keys and unique keys.
j. MySQL has a speed in making tables and updating tables

1.2.3. Get to Know Database

Database is a collection of connected data that is stored together on a media, which is organized based on specific schema or structure, and with software to manipulate for certain uses. Basic database operations:

a. Create database
b. Drop database
c. Create table
d. Drop table
e. Insert
f. Retrieve / Search
g. Update

h. Delete

1.2.4. Case Study : Library Management System

When borrowing books in a library, someone will register first. Then, the borrower's data will be stored by the system in the database. After that, various transactions can be carried out, e.g. book lending, book lending extension, book lending extension, book repayment, and payment of fines. All transactions carried out are stored in a database and can be accessed again by the system as needed.

Borrowers can search for books to borrow by pressing the find or search button. After that the information system will interact with the library database to search for the book. Then the search results will be displayed on the screen. Through the interaction that occurs between the information system and the database, library staff can find out automatically if the book lending is more than the applicable provisions, the borrower has not returned the book but will borrow another book, and notification that there is a fine on the D-Day. stored in a database include:

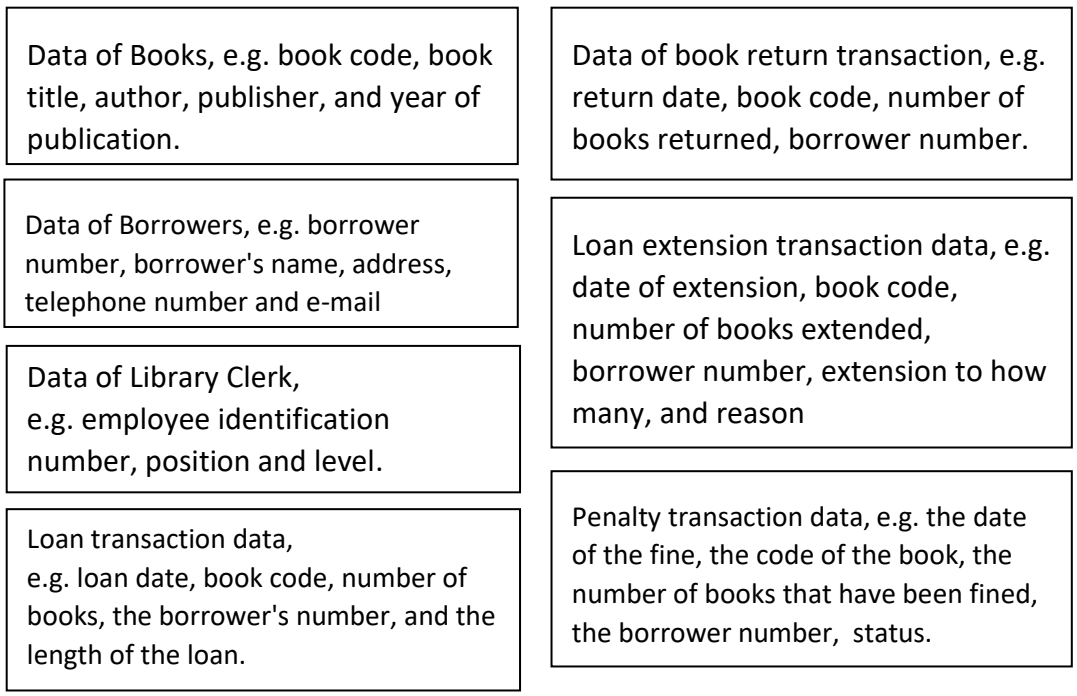


Figure 1.1 Example of Data in Library System

1.3. TOOLS

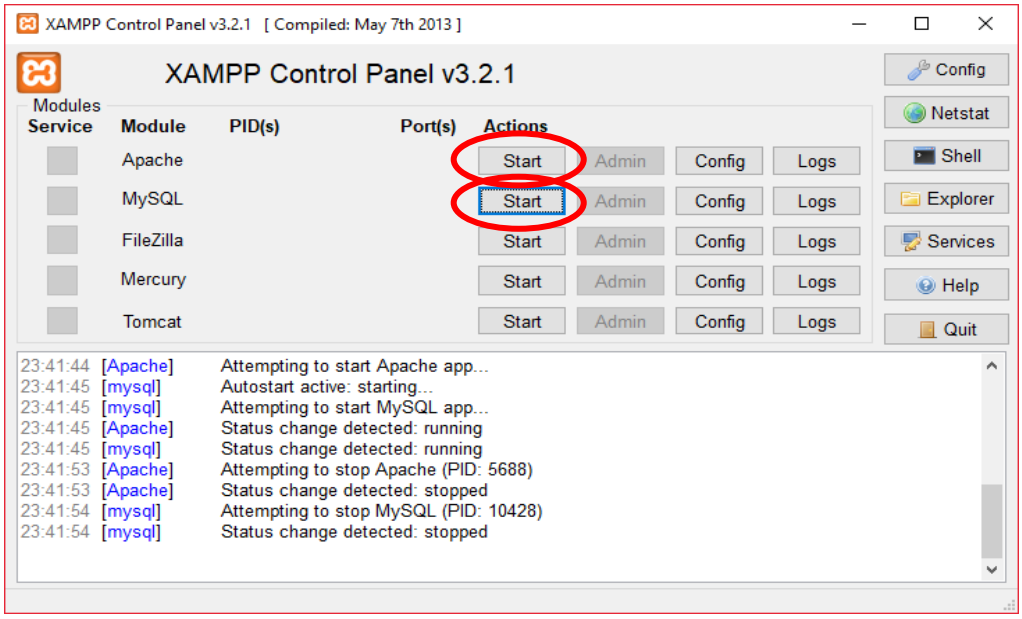
Material used in this chapter:

1. Computer
2. XAMPP

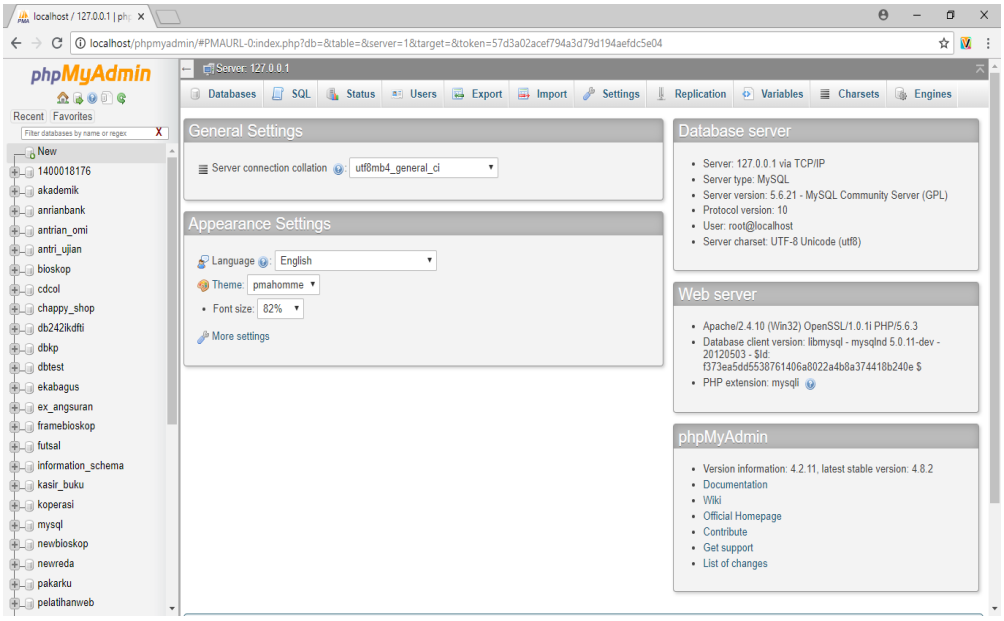
1.4. PRACTICUM INSTRUCTIONS

1.4.1. XAMPP to Access PHP MyAdmin

Run XAMPP Control Panel until the XAMPP Control Panel application window appears. Then click the Start button on the Apache and MySQL modules. Apache module is used to access PHPMyAdmin on the browser so it is easier to access MySQL because it uses a GUI (Graphical User Interface). The MySQL module is used to serve requests or queries received from PHPMyAdmin.

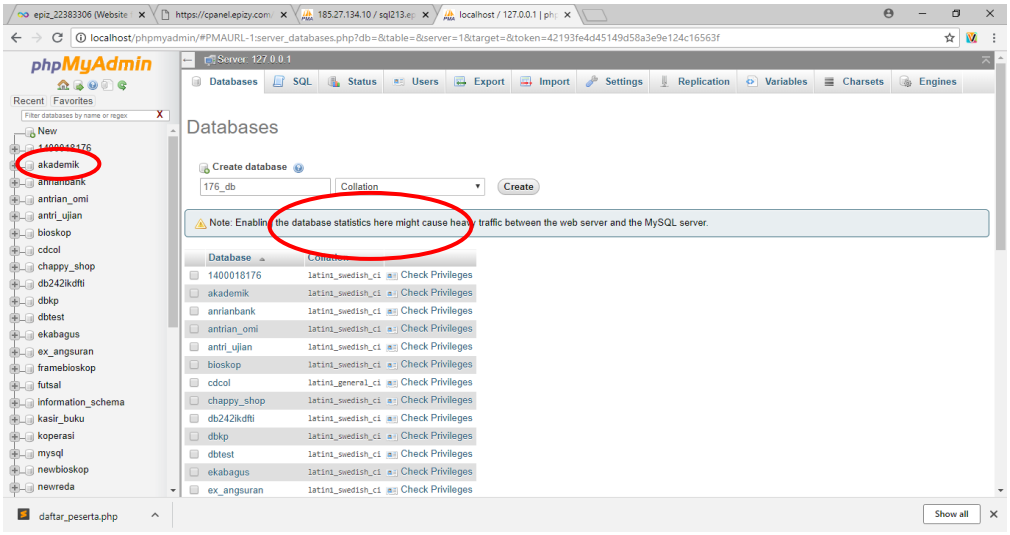


Then open the browser (Chrome, Mozilla, Opera, etc.) and access it by typing "localhost / phpmyadmin" in the URL field, so that it appears like in the picture below.



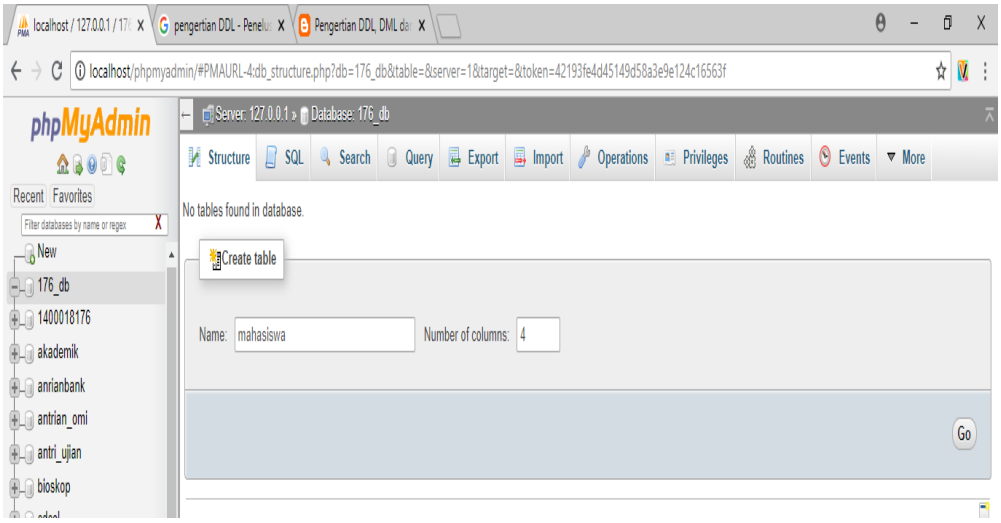
1.4.2. Creating Database

Select the "new" menu then fill in the database name in the column provided, for uniformity the database name is filled with "3_digit_nim_terakhir_db" then select "create".

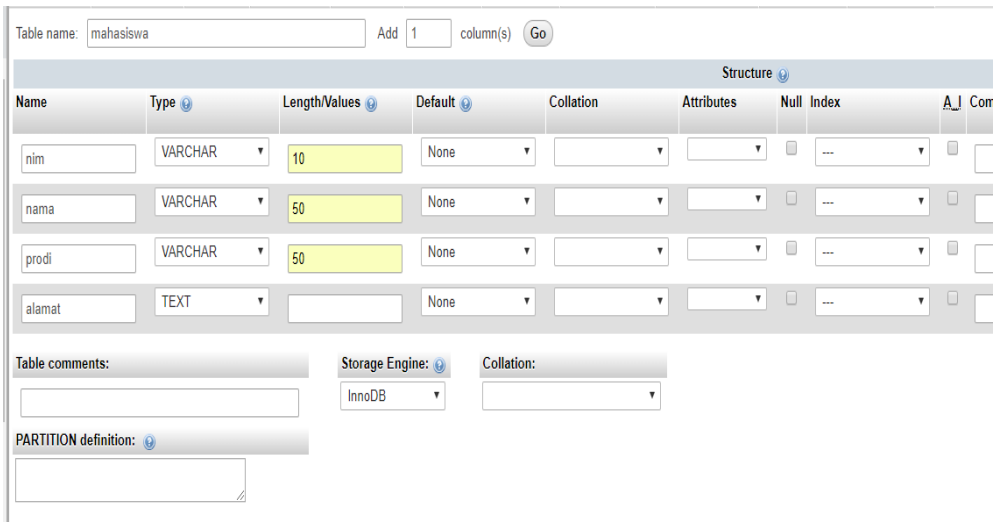


1.4.3. Creating Tables

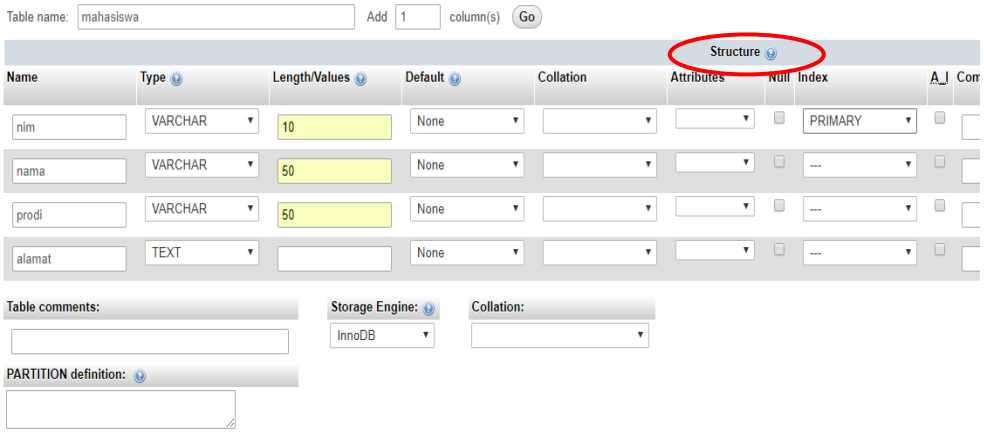
Make a table with the table name "student" that has nim attributes, names, study programs and addresses. The steps are to select the database menu until the image appears as below.



Then fill in the form to give the attribute to the table "student" that has been made. In this entity, the nim attribute will be used as a differentiator for data that will be entered into a table or often called a primary key. The attribute nim, name and study program are identified by the VARCHAR type with a character length that is adjusted to the wishes of the programmer. While the address attribute is identified by the TEXT type because it has a longer number of characters.



Then in the attribute nim, because as a primary key the index is filled with PRIMARY then select "Save".

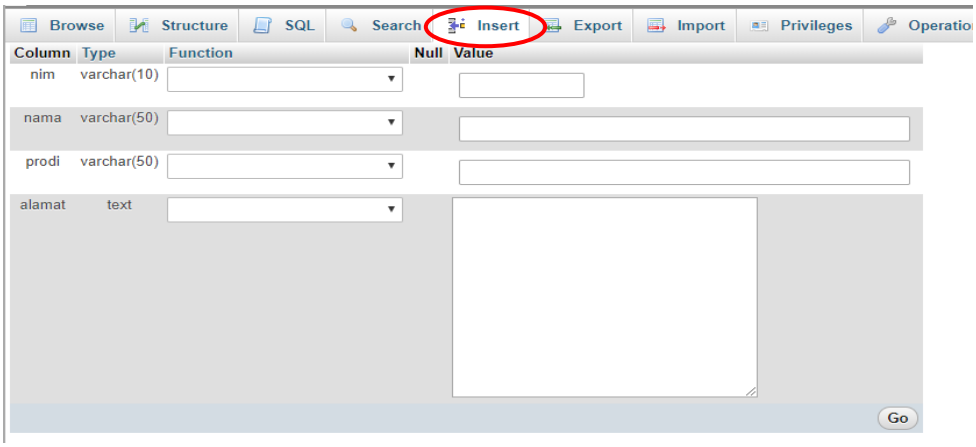


1.4.4. Insert Data

This step is used to input data into the "student" table, for example 3 data will be inputted.

The steps to add data are as follows:

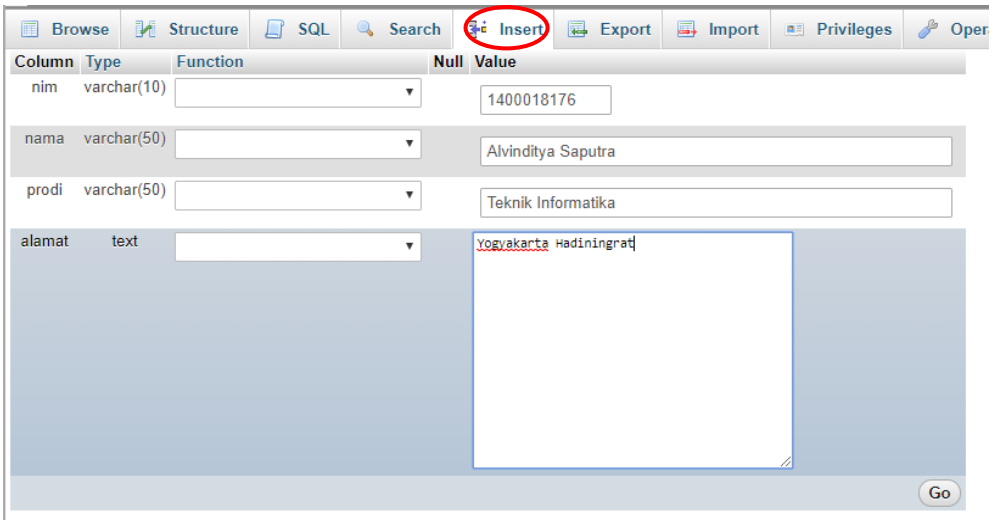
a. Select the "Insert" menu so that it appears as in the image below



a. Then fill in the fields or fields in the value section then select "Go", as in the image below.

b. To check whether the data has entered the database, you can do this by selecting on the

"Browse" menu. Like in the picture below



Repeat the steps for entering data until there are 3 data in the "student" table. Please note that the attribute nim is the primary key, so the values entered cannot be the same.

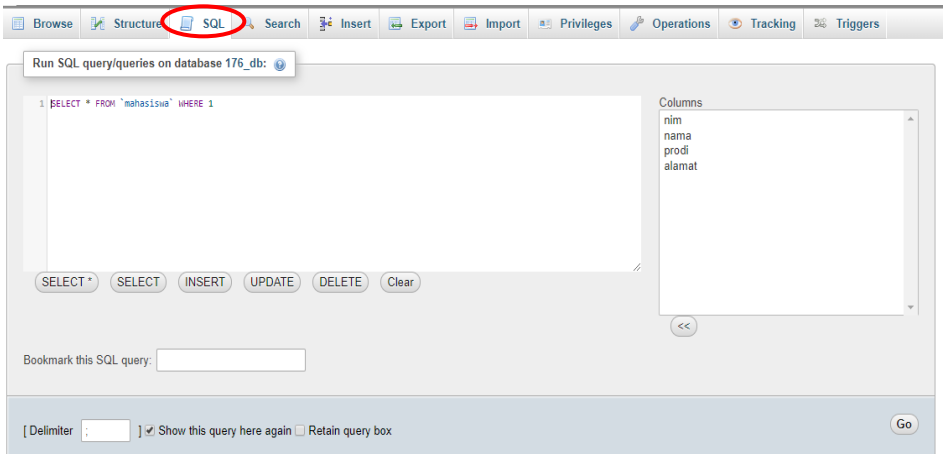
1.4.5. Read Data

This step is used to read or retrieve data contained in student tables. There are several queries to read the data according to what you want displayed.

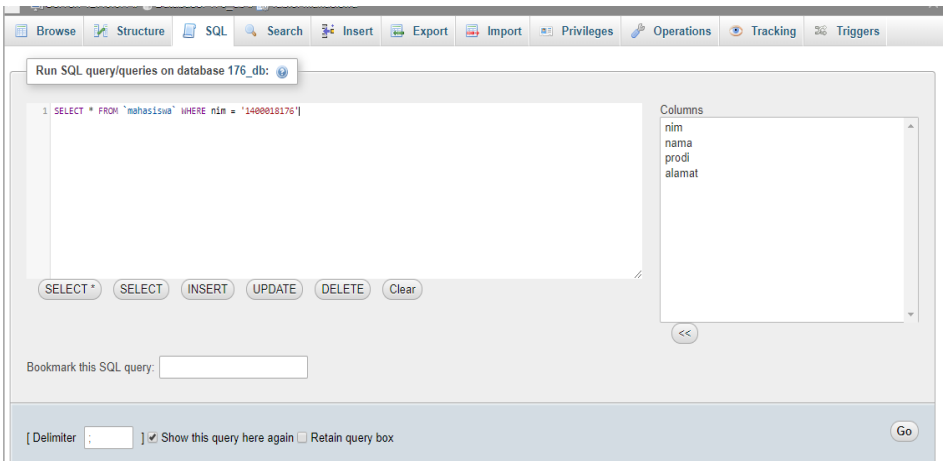
a. SELECT *: To read data with all attributes displayed. Query: SELECT * FROM table_name.
b. SELECT: To read data with one or more attributes displayed. Query: SELECT attribute_1, attribute_2 FROM table_name.
c. SELECT ... WHERE: To read data with a desired condition. Query: SELECT * FROM table_name WHERE id = 1.

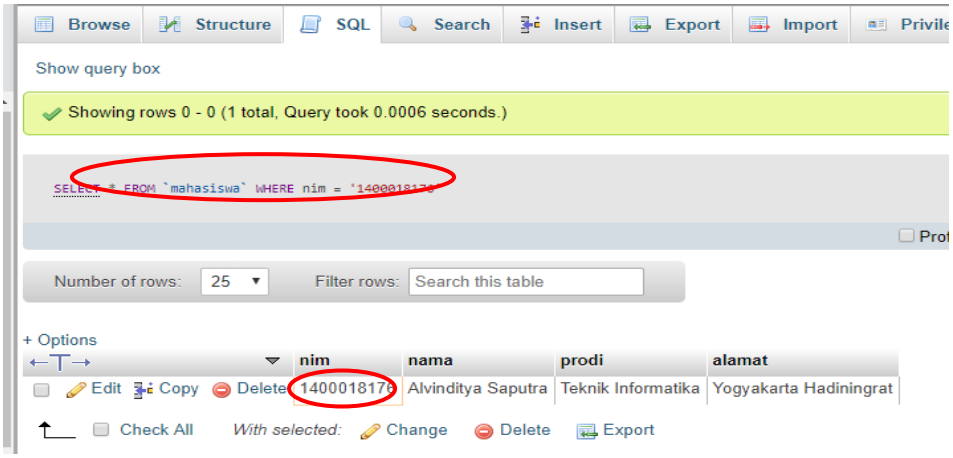
The steps to display data are as follows:

a. Select the SQL menu on PHPMyAdmin so it looks like the image below



b. For example, the data displayed is student with student number 1400018176. Then the query needed is SELECT * FROM student WHERE student = = 1400018176 '. Then select "Go", so that it looks like in the image below.

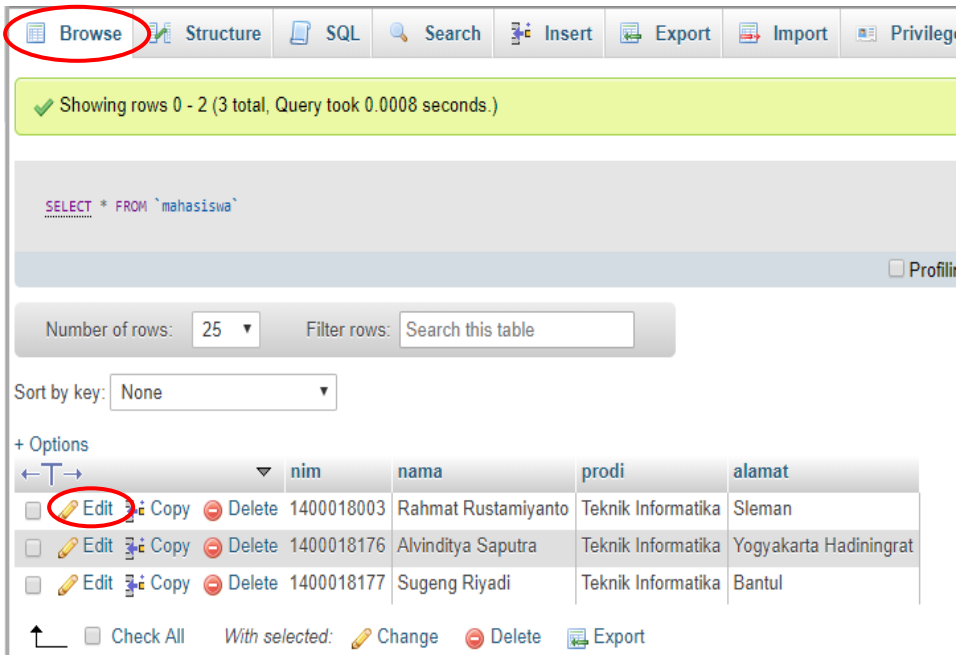




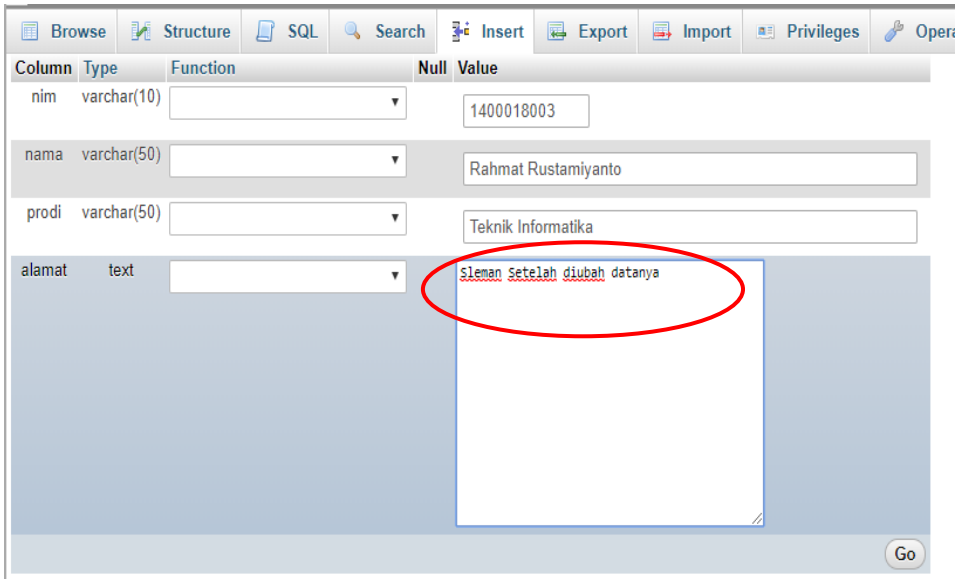
1.4.6. Update Data

This step is used to change or modify the attribute or data values in a table or entity. The steps to change data or update data are as follows:

a. Make a table with the table name "student" that has nim attributes, names, study programs and addresses. The steps are to select the database menu until the image appears as below.

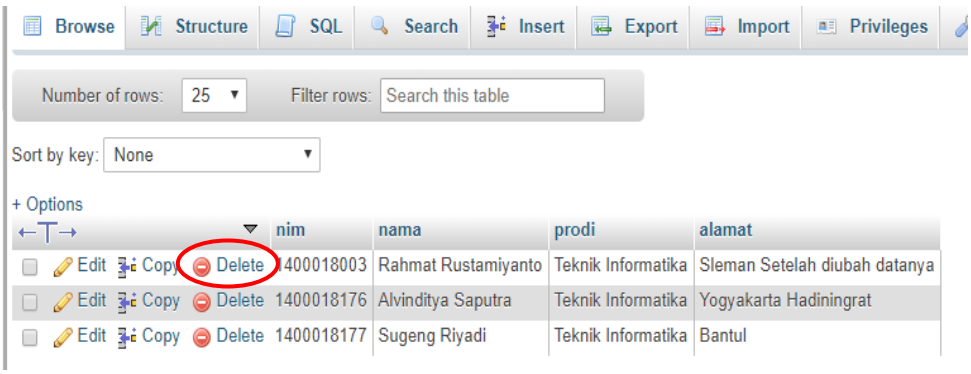


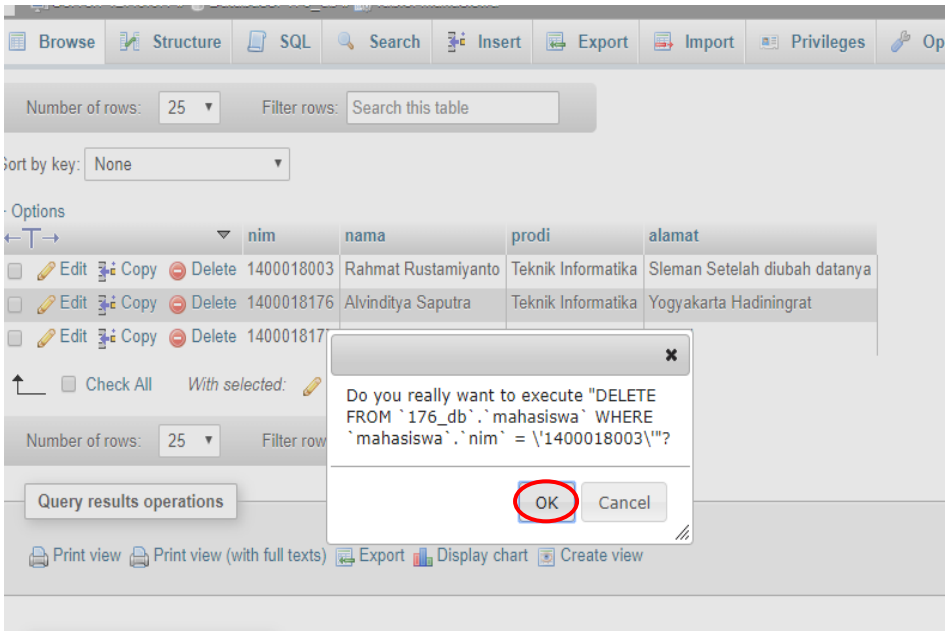
b. Then fill in the fields to be changed, for example the data to be changed is in the address field. Then click "Go" so that it looks like the image below.

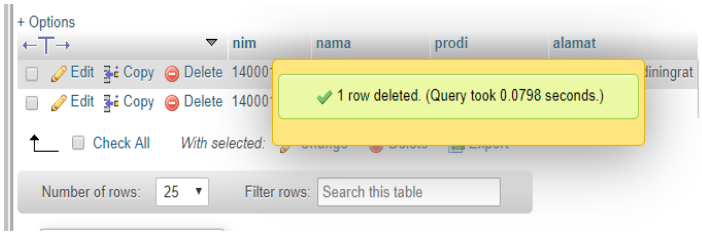


1.4.7. Delete data

In this step the DELETE command is used to delete one or more data in a table. Steps to delete data is to click "Delete" on the "Browse" menu as shown in the picture below.







1.5. EXERCISES

The assignment contains a post test that must be done by students as an evaluation of the practicum done (example evaluation sheet attached).

EXAMPLE OF
PRE-TEST / POST-TEST / EVALUATION ANSWER SHEET OF PRACTICUM 1: PRAKTIKUM NAME

Name :	**Pract Assistant:**	**Date:**
Student Number :	**Signature:**	**Score:**

MEETING 2: RELATION TABLE :MULTIPLICITY AND CARDINALITY

Meeting : 2

Time Allocation : 150 minutes
- Pre-Test : 30 minutes
- Prakticum : 90 minutes
- Post-Test : 30 minutes

Scoring : 100%
- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

2.1. GOALS AND PERFORMANCE INDICATORS

By following this session, students are expected:

1. Able to get to know MySQL and Database Management System.
2. Able to understand table, data type and primary key within Database Management System.
3. Able to identify entity from case(s) provided.
4. Able to process data in a table which includes adding data, deleting data and editing data

Performance indicators are measured by:

1. Insert, Update, Delete

2.2. STUDY LITERATURE

2.2.1. Relation

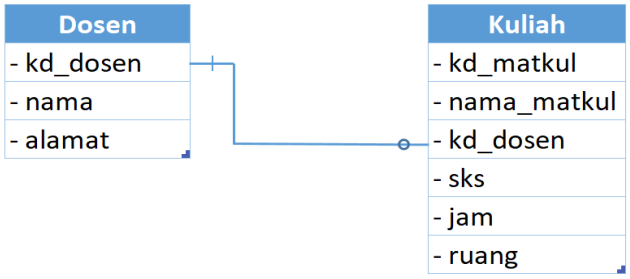
Relationships between tables that present relationships between objects in the real world. Relationships are relationships that occur in a table with others that present relationships between objects in the real world and function to regulate the operation of a database. Relationships that can be formed can include 3 types of relationships:

a. One to One

Having the understanding "Each row of data in the first table is connected to only one row of data in the second table". For example: A student has exactly one student number.

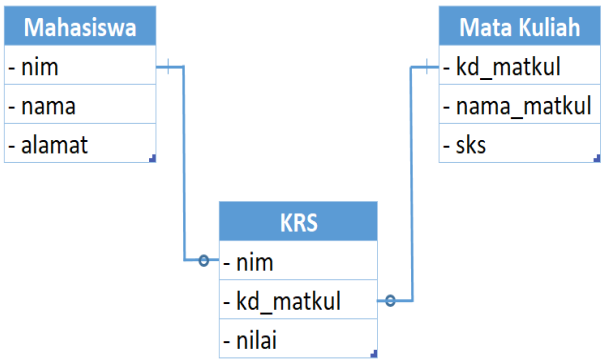
b. One to Many

Having the understanding "Each row of data from the first table can be connected to one or more rows of data in the second table". For example: A lecturer teaches many subjects.



c. Many to Many

Having the understanding "One or more rows of data in the first table can be connected to one or more rows of data in the second table". This means that there are many rows in table one and table two that are interconnected with each other. For example: Many Students take many courses.



2.2.2. Primary Key dan Foreign Key

Primary key is the key / main field of a table which shows that the key field cannot be filled with the same data, / in other words Primary key makes each record has its own identity that distinguishes one another (unique). Whereas a foreign key is an attribute (or a set of attributes) that complements a relationship that points to its parent. Foreign key is used to define columns in a table whose values refer to other tables, so foreign key columns must be taken from the value of columns in other tables.

2.3. TOOLS

Material used in this chapter:

1. Computer
2. XAMPP

2.4. PRACTICUM INSTRUCTION

2.4.1. Main Table

Create a "Lecturer" table with the code_dsn, nama_dsn, and address_dsn attributes and create a table "Mata_Kuliah" with the attributes Kode_mk, nama_mk, credits. Then fill in the data in each table 3 data as in the previous meeting.

	kode_dsn	nama_dsn	alamat_dsn
Edit Copy Delete	50001	Slamet Widodo	Yogyakarta
Edit Copy Delete	50002	Wahyu Subrata	Bandung
Edit Copy Delete	50003	Sri Linggajati	Lampung

	kode_mk	nama_mk	sks
Edit Copy Delete	T1001	Basis Data	4
Edit Copy Delete	T1002	Pemrograman Web	3
Edit Copy Delete	T1003	Statistika	4

2.4.2. Relation One-to-One Tables

Create a "Class" table to implement the One to One type of relationship, as shown below.

Table name: kelas Add 1 column(s) Go

Structure

Name	Type	Length/Values	Default	Collation	Attributes	Null	Index
kode_kelas	VARCHAR	10	None				PRIMARY
angkatan	VARCHAR	50	None				---
nim_ketua	VARCHAR	10	None				---
	INT		None				---

Then inputting data with the records must already exist in the student table, because this table will read data information from the student table as in the example below.

+ Options

	kode_kelas	angkatan	nim_ketua
Edit Copy Delete	14_C	2014	1400018176

Check All With selected: Change Delete Export

2.5. EXERCISES

The assignment contains a post test that must be done by students as an evaluation of the practicum done (example evaluation sheet attached).

EXAMPLE OF

PRE-TEST / POST-TEST / EVALUATION ANSWER SHEET OF PRACTICUM 3: PRAKTICUM NAME

Name :	**Pract Assistant:**	**Date:**
Student Number :	**Signature:**	**Score:**

MEETING 3: PHP and MySQL

Meeting : 3

Time Allocation : 150 minutes

- Pre-Test : 30 minutes
- Practicum : 60 minutes
- Post-Test : 30 minutes

Scoring : 100%

- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

3.1. GOALS AND PERFORMANCE INDICATORS

By following this session, students are expected:

1. Students able to make connections to the database using PHP Objects.
2. Students able to make CRUD queries on the database using PHP Objects.

Performance indicators are measured by:

1. MySQL, PHP, HTML

3.2. STUDY LITERATURE

PHP is a server-side script programming language designed for web development. In addition, PHP can also be used as a general programming language. PHP is called the server-side programming language because PHP is processed on the server computer. This is different compared to client-side programming languages like JavaScript that are processed in a web browser (client).

Object is a collection of software consisting of variables and related methods. Objects are also real objects that are created based on the design defined in the class. Object is an instance of class. If a class generally represents a template, an instance is a real representation of the class itself.

Class is a prototype, or blueprint, or design that defines variables and methods for all specific objects. Class functions to accommodate the contents of the program to be run, in which it contains attributes / data types and methods for running a program. While the method is an operation in the form of functions that can be done by an object. The method is defined in the class but is called via the object.

3.3. TOOLS

Material used in this chapter:

1. Computer
2. XAMPP
3. Web Browser

3.4. PRACTICUM INSTRUCTION

3.4.1. Create Class in PHP

Create a folder located in "C: \\ xampp \ htdocs \" which is used to access php files in it through a web browser. Create a php file that will be used as a place to create classes in PHP, for example with the name 'database.php'.

```
<?php
Class Database{
        function __construct(){
                $this->db = new mysqli("localhost", "root", "", "176_db");
        }
}
?>
```

3.4.2. Displaying Student Data

Create a function that is used to display student data:

```
function tampilMhs(){
                $array = array();
                $query = $this->db->query("SELECT * FROM mahasiswa");
```

```
            while ($data = mysqli_fetch_array($query)){
                $array[] = $data;
            }
            return $array;
        }
```

Then create an index.php file that will be used to display data from the Appeals function onto the web.

```
<?php
    require 'database.php';
    $objek = new Database();
    $dataMhs = $objek->tampilMhs();
?>
```

The source code above functions to call the database.php file and then create an object from the Database class. Then create HTML and PHP source code to display the results of the data to be displayed.

```
<table border="1">
    <tr>
        <th>NIM</th>
        <th>Nama</th>
        <th>Program Studi</th>
        <th>Alamat</th>
    </tr>
    <?php foreach ($dataMhs as $x) {?>
    <tr>
        <td><?php echo $x['nim'];?></td>
```

```
                <td><?php echo $x['nama'];?></td>
                <td><?php echo $x['prodi'];?></td>
                <td><?php echo $x['alamat'];?></td>
        </tr>
        <?php } ?>
</table>
```

The foreach function is used to break the array of data returned by the tampilMhs function in the Database class.

NIM	Nama	Program Studi	Alamat
1400018176	Alvinditya Saputra	Teknik Informatika	Yogyakarta Hadiningrat
1400018177	Sugeng Riyadi	Teknik Informatika	Bantul

3.4.3. Insert Student Data

The first thing to do is create a student input form by adding the source code below to index.php.

```
<form method="POST" action="">
        <table>
                <tr>
                        <td>NIM</td>
                        <td>:</td>
                        <td><input type="text" name="nim"></td>
                </tr>
                <tr>
                        <td>Alamat</td>
                        <td>:</td>
                        <td><input type="text" name="nama"></td>
                </tr>
```

```
                <tr>
                        <td>Program Studi</td>
                        <td>:</td>
                        <td><input type="text" name="prodi"></td>
                </tr>
                <tr>
                        <td>Alamat</td>
                        <td>:</td>
                        <td><input type="text" name="alamat"></td>
                </tr>
                <tr><td colspan="3" align="right"><input type="submit"
name="simpan" value="SIMPAN"></td></tr>
        </table>
</form>
```

NIM	Nama	Program Studi	Alamat
1400018176	Alvinditya Saputra	Teknik Informatika	Yogyakarta Hadiningrat
1400018177	Sugeng Riyadi	Teknik Informatika	Bantul

NIM :
Alamat :
Program Studi :
Alamat :
SIMPAN

Then create a function in the database class that functions to input data into the database.

```
function tambahMhs($nim, $nama, $prodi, $alamat){
                $insert = $this->db->query("INSERT INTO mahasiswa (nim, nama,
prodi, alamat) VALUES ('$nim','$nama','$prodi','$alamat')");
```

```
            if ($insert) {
                header('location:index.php');
            }else{
                echo "data gagal ditambahkan";
                echo "<br>$nim $nama $prodi $alamat";
                echo $this->db->error;
            }
        }
```

Then add the source code to the php index which functions to deliver data to the function that was created in the previous step.

```
<?php
    if (isset($_POST['simpan'])) {
        $nim = $_POST['nim'];
        $nama = $_POST['nama'];
        $prodi = $_POST['prodi'];
        $alamat = $_POST['alamat'];
        $objek->tambahMhs($nim, $nama, $prodi, $alamat);
    }
?>
```

NIM	Nama	Program Studi	Alamat
1400018176	Alvinditya Saputra	Teknik Informatika	Yogyakarta Hadiningrat
1400018177	Sugeng Riyadi	Teknik Informatika	Bantul

NIM : 1400018182
Alamat : Andi Eko
Program Studi : Teknik Informatika
Alamat : Lampung
SIMPAN

3.4.4. Update and Delete Student Data

Change the html syntax on index.php which displays student data tables, as seen below.

```
<tr>
                <th>NIM</th>
                <th>Nama</th>
                <th>Program Studi</th>
                <th>Alamat</th>
                <th>Aksi</th>
        </tr>
        <?php foreach ($dataMhs as $x) {?>
        <tr>
                <td><?php echo $x['nim'];?></td>
                <td><?php echo $x['nama'];?></td>
                <td><?php echo $x['prodi'];?></td>
                <td><?php echo $x['alamat'];?></td>
                <td><a href="edit.php?nim=<?php echo $x['nim'];?>">Edit</a> |
<a href="index.php?aksi=delete&nim=<?php echo
$x['nim'];?>">Hapus</a></td>
        </tr>
```

Then make an update function on the Database class.

```
function updateMhs($nim, $nama, $prodi, $alamat){
        $insert = $this->db->query("UPDATE mahasiswa SET nama =
'$nama', prodi = '$prodi', alamat = '$alamat' WHERE nim = '$nim'");
        if ($insert) {
            header('location:index.php');
        }else{
            echo "data gagal ditambahkan";
            echo "<br>$nim $nama $prodi $alamat";
            echo $this->db->error;
        }
    }
```

Then create an edit.php file with html and php syntax as shown below.

```
<?php
    require 'database.php';
    $objek = new Database();
    $mhs = $objek->tampilMhsWhere($_GET['nim']);
    foreach ($mhs as $x) {
    }
    if (isset($_POST['simpan'])) {
        $nim = $_POST['nim'];
        $nama = $_POST['nama'];
        $prodi = $_POST['prodi'];
        $alamat = $_POST['alamat'];
        $objek->updateMhs($nim, $nama, $prodi, $alamat);
```

```
        }
?>
<form method="POST" action="">
        <table>
                <tr>
                        <td>NIM</td>
                        <td>:</td>
                        <td><input type="text" name="nim" value="<?php echo $x['nim'];?>" readonly></td>
                </tr>
                <tr>
                        <td>Alamat</td>
                        <td>:</td>
                        <td><input type="text" name="nama" value="<?php echo $x['nama'];?>"></td>
                </tr>
                <tr>
                        <td>Program Studi</td>
                        <td>:</td>
                        <td><input type="text" name="prodi" value="<?php echo $x['prodi'];?>"></td>
                </tr>
                <tr>
                        <td>Alamat</td>
                        <td>:</td>
```

```
                    <td><input type="text" name="alamat" value="<?php echo
$x['alamat'];?>"></td>

                </tr>

                <tr><td colspan="3" align="right"><input type="submit"
name="simpan" value="SIMPAN"></td></tr>

            </table>

</form>
```

NIM	Nama	Program Studi	Alamat	Aksi
1400018176	Alvinditya Saputra	Teknik Informatika	Yogyakarta Hadiningrat	Edit | Hapus
1400018177	Sugeng Riyadi	Teknik Informatika	Bantul	Edit | Hapus
1400018182	Andi Eko	Teknik Informatika	Lampung	Edit | Hapus

NIM :
Alamat :
Program Studi :
Alamat :
SIMPAN

NIM	Nama	Program Studi	Alamat	Aksi
1400018176	Alvinditya Saputra	Teknik Informatika	Yogyakarta Hadiningrat	Edit | Hapus
1400018177	Sugeng Riyadi	Teknik Informatika	Bantul	Edit | Hapus
1400018182	Andi Eko Suryanto	Teknik Informatika	Lampung	Edit | Hapus

NIM :
Alamat :
Program Studi :
Alamat :
SIMPAN

Then create a delete function on the database class that functions to delete data based on nim.

```
function hapusMhs($nim){

                $insert = $this->db->query("DELETE FROM mahasiswa WHERE
nim = '$nim'");
```

```
            if ($insert) {
                header('location:index.php');
            }else{
                echo "data gagal dihapus";
                echo "<br>$nim";
                echo $this->db->error;
            }
        }
```

Then add the php source code to index.php as below.

```
if (isset($_GET['aksi'])) {
            if ($_GET['aksi'] == 'delete') {
                $objek->deleteMhs($_GET['nim']);
            }
        }
```

NIM	Nama	Program Studi	Alamat	Aksi
1400018176	Alvinditya Saputra	Teknik Informatika	Yogyakarta Hadiningrat	Edit | Hapus
1400018177	Sugeng Riyadi	Teknik Informatika	Bantul	Edit | Hapus
1400018182	Andi Eko Suryanto	Teknik Informatika	Lampung	Edit | Hapus

NIM :
Alamat :
Program Studi :
Alamat :
SIMPAN

NIM	Nama	Program Studi	Alamat	Aksi
1400018176	Alvinditya Saputra	Teknik Informatika	Yogyakarta Hadiningrat	Edit | Hapus
1400018177	Sugeng Riyadi	Teknik Informatika	Bantul	Edit | Hapus

3.5. EXERCISES

The assignment contains a post test that must be done by students as an evaluation of the practicum done (example evaluation sheet attached).

EXAMPLE OF
PRE-TEST / POST-TEST / EVALUATION ANSWER SHEET OF PRACTICUM 3: PRAKTICUM NAME

Name :	**Pract Assistant:**	**Date:**
Student Number :	**Signature:**	**Score:**

MEETING 4: NORMALIZATION

Meeting : 4

Time Allocation : 150 minutes

- Pre-Test : 30 minutes
- Practicum : 60 minutes
- Post-Test : 30 minutes

Scoring : 100%

- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

4.1. GOALS AND PERFORMANCE INDICATORS

By following this session, students are expected:

1. Able to make understand DDL and DML commands

Performance indicators are measured by:

1. Create, Alter, Drop, Insert, Update, Delete, Select

4.2. STUDY LITERATURE

Normalization process is the process of grouping data elements into tables showing the entities and their relationships. Normalization process can also be defined as the process of creating a table (relation) in the database in order to reduce redundancy.

In the normalization process, it is always tested under several conditions. Are there any difficulties when adding / inserting, deleting / deleting, modifying / updating, reading / retrieving a database. If there are difficulties in the test, the relationship is solved in several tables again, or in other words, the total design is to get the optimal database

In normalization, the term anomaly is known which means the problems that arise in making tables. Anomaly is a process in the database that has unexpected side effects. For example: data inconsistency, lost data when deleted, etc.

Before getting to know more about normalization, there are several concepts that must be known first, namely:

a. *Table Attribute*

The true attributes are synonymous with the term data column. The term attribute is more commonly used in database design, because the word is more impressive in showing its function as a forming characteristic attached to a table.

Besides the unique naming based on their function in each table, the attributes can be differentiated based on a number of groupings. Some attributes are used as keys and others are called descriptive attributes. There are also attributes that are classified as simple attributes or composite attributes, and so on

Key

Every file always has a key from the file in the form of a field or a set of fields that can represent a record. For example, the employee number is the key to the employee table of a company, every search is sufficient to mention the employee's number so that the name, address and other attributes of an employee can be found. There are three kinds of keys that can be applied to a table, namely:

1. *Superkey*: is one or more attributes that can uniquely distinguish each row of data in a table.
2. *Candidate Key*: is a minimal set of attributes that can uniquely distinguish each row of data in a table. A candidate-key cannot contain an attribute or set of attributes that has become another superkey.
3. *Primary Key*: is an attribute or a minimum set of attributes that not only uniquely identify a specific event, but also can represent every occurrence of an entity.

b. Descriptive attributes

Descriptive attributes are attributes that are not a key or an attribute that is a member of the primary key.

c. functional Dependency

Given a relation R, the Y attribute of R is a function dependent on the X attribute of R if and only if each of the X values in R has a relationship with exactly one Y value in R (at any one time).

The employee relation table contains attributes : No_Induk, No_KTP, Nama, Tempat_Lahir, Tanggal_Lahir, Alamat, Kota.

The contents of the Name attribute depend on Nomor_Induk. So it can be said that the attribute Name is functionally dependent on No_Induk dan No_Induk indicates by function Name. If you know No_Induk pegawai. then you can specify the employee's name. The notation for the dependency of this function is:

No_Induk -> Nama
Atau
Nama = f(No_Induk)

4.3. PRACTICUM INSTRUCTION

A. Normalization steps

Case study:

There is a *Kartu Hasil* which still in a manual and we will analyze it using the Normalization technique as follows *:*

KARTU HASIL

NIM : 061
Nama Mahasiswa : Anto
Jurusan : Teknik Informatika
Fakultas : FTI

No.	Kode	Nama Mata Kuliah	Kode Dosen	Ruang	Nilai
1.	001	Algoritma Pemrograman	D01	B431	B
2.	002	Pemrograman Web	D02	B432	A

KARTU HASIL

NIM : 078
Nama Mahasiswa : Hanifah
Jurusan : Teknik Informatika
Fakultas : FTI

No.	Kode	Nama Mata Kuliah	Kode Dosen	Ruang	Nilai
1.	001	Algoritma Pemrograman	D01	B431	B
2.	002	Pemrograman Web	D02	B432	A
3.	015	Praktikum P. Web	D10	B461	B

figure 1.31. Kartu Hasil

1. Unnormalized Form

This form is a collection of data to be recorded, there is no need to follow a certain format, the data may be incomplete or duplicated. Data is collected as is according to arrival.

Abnormal form means a data set to be processed which is obtained from various formats, still duplicates, may be imperfect or incomplete, and according to field facts. This form is obtained from existing documents in the field or manual with attributes not simple values.

NIM	Nama_Mahasiswa	Jurusan	Fakultas	Kode_Matkul	Nama_Matkul	ID_Dosen	Ruang	Nilai
061	Anto	Teknik Informatika	FTI	001	Algoritma Pemrograman	D01	B431	B
				002	Pemrograman Web	D02	B432	A
078	Hanifah	Teknik Informatika	FTI	001	Algoritma Pemrograman	D01	B431	B
				002	Pemrograman Web	D02	B432	A
				015	Praktikum P. Web	D10	B461	B

2. 1st Normal Form (1NF)

The first normal form has characteristics, namely that each data is formed in a flat file, the data is formed in one record by one record and the value of the field is an atomic value. No attribute set is repeated or multiple values. Sutau table is considered normal to one (1NF) if:

1) There are no duplicate or duplicate rows
2) Each row is single and not null

Step-by-step:

1) Fill in every single data value and not null
2) Discard looping data in another row

NIM	Nama_Mahasiswa	Jurusan	Fakultas	Kode_Matkul	Nama_Matkul	ID_Dosen	Ruang	Nilai
061	Anto	Teknik Informatika	FTI	001	Algoritma Pemrograman	D01	B431	B
061	Anto	Teknik Informatika	FTI	002	Pemrograman Web	D02	B432	A
078	Hanifah	Teknik Informatika	FTI	001	Algoritma Pemrograman	D01	B431	B
078	Hanifah	Teknik Informatika	FTI	002	Pemrograman Web	D02	B432	A
078	Hanifah	Teknik Informatika	FTI	015	Praktikum P. Web	D10	B461	B

Gambar 1.33. Data Mahasiswa 2

3. 2nd Normal Form (2NF)

Bentuk The second normal form (2NF) is fulfilled if:

1) Has first normal form (1NF)
2) All attributes that are not included in the primary key have a functional dependence on the primary key as a whole. An attribute is said to have a functional dependency if the attribute's price determines the price of another attribute. For example:

NIM -> Nama_Mahasiswa

tepse:

If there are attributes that depend on non-primary key attributes and are key attributes, it will become a new table.

Tabel 1.10. Mahasiswa

NIM	Nama_Mahasiswa	Jurusan	Fakultas
061	Anto	Teknik Informatika	FTI
078	Hanifah	Teknik Informatika	FTI

Tabel 1.11. Mata Kuliah

Kode_Matkul	Nama_Matakuliah	ID_Dosen
001	Algoritma Pemrograman	D01
002	Pemrograman Web	D02
015	Praktikum P. Web	D10

Tabel 1.12. Nilai

NIM (fk)	Kode_Matkul (fk)	Ruang	Nilai
061	01	B431	B

061	02	B432	A
078	01	B431	B
078	02	B432	A
078	015	B461	B

4. 3rd Normal Form (3NF)

Database normalization in the form of 3NF aims to eliminate all attributes or fields that are not related to the primary key. Thus there is no transitive dependence on each candidate key.

Third Normal Form (3NF) is fulfilled if:

1) Has Second normal form (2NF)
2) There are no anomalies resulting from transitive dependence. Transitive dependencies are functional dependecies between 2 or more non-key attributes.

Steps :

1) Make sure all non-key attibutes fully depend on the key attribute
2) Separate into a new table if you find transitive dependencies in that table.

Tabel Matakuliah

Kode_Matkul	**Nama_Matakuliah**	**ID_Dosen (fk)**
01	Algoritma Pemrograman	D01
02	Pemrograman Web	D02
015	Praktikum Pemrograman Web	D10

Tabel Dosen

ID_Dosen	**Nama_Matakuliah**
D01	Budiyanto M.Kom
D02	Ratna MCs
D10	Ardi M.T.

5. Boyce-Codd Normal Form (BCNF)

Practically the objective of database analysis is enough to reach 3NF. However, in a certain case it is better to get BCNF. Some thinkers equate 3NF with BCNF. BCNF has a stronger force than the 3NF form.

The normal form of BCNF is fulfilled if:

Each of the main attributes fully functional on each key, where the key is not part of it.

1) Every determinant of relation attributes is the relation key or candidate key.
2) BCNF can have more than one key.
3) BCNF is almost the same as 3NF.

Steps :

1) remove dependencies on non-candidate keys

6. 4th Normal Form (4NF) dan 5th Normal Form (5NF)

The application of normalization rules up to the third stage is actually very adequate to produce good quality tables. However, from a number of literatures there is also discussion about the fourth normal form (4NF) and the fifth normal form (5NF). The fourth normal form relates to the multivalued dependency on a table which is a development of functional dependency. Meanwhile, the fifth normal form relates to the dependency of relations between tables (join dependency). The discussion of the last two normal forms is quite complex, but the benefits themselves are not that great.

4.4. EXERCISES

The assignment contains a post test that must be done by students as an evaluation of the practicum done (example evaluation sheet attached).

EXAMPLE OF
PRE-TEST / POST-TEST / EVALUATION ANSWER SHEET OF PRACTICUM 3: PRACTICUM NAME

Name :	**Pract Assistant:**	**Date:**
Student Number :	**Signature:**	**Score:**

MEETING 5: DDL and DML

Meeting : 5

Time Allocation : 150 minutes

- Pre-Test : 30 minutes
- Practicum : 60 minutes
- Post-Test : 30 minutes

Scoring : 100%

- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

5.1. GOALS AND PERFORMANCE INDICATORS

By following this session, students are expected:

2. Able to make understand DDL and DML commands

Performance indicators are measured by:

2. Create, Alter, Drop, Insert, Update, Delete, Select

5.2. STUDY LITERATURE

Data Definition Language (DDL) has functions to do the following things:

1. Creating / deleting databases, declared with the CREATE DATABASE and DROP DATABASE commands
2. Creating / deleting tables, declared with the CREATE TABLE and DROP TABLE commands
3. Modifying the table, declared with the ALTER TABLE command

While the Data Manipulation Language (DML) consists of:

1. Fill the table with data, declared with the INSERT command
2. Editing data in a table, declared with the UPDATE command
3. Deleting data in a table, declared by the DELETE command
4. Looking for data in a table, expressed by the SELECT command.

5.3. TOOLS

Material used in this chapter:

1. Computer
2. XAMPP
3. Web Browser

5.4. PRACTICUM INSTRUCTION

5.4.1. DDL Commands

1. Display the existing database on MySQL

   SHOW databases;

2. Create database

   CREATE database <database_name>;

   CREATE database akademik;

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| test               |
+--------------------+
3 rows in set (0.02 sec)

mysql> create database akademik;
Query OK, 1 row affected (0.00 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| akademik           |
| mysql              |
| test               |
+--------------------+
4 rows in set (0.00 sec)
```

3. Use database

   USE <database_name>;

   USE akademik;

4. Delete database

   DROP < database_name >;

   DROP akademik;

```
+--------------------+
| Database           |
+--------------------+
| information_schema |
| akademik           |
| mysql              |
| test               |
+--------------------+
4 rows in set (0.00 sec)

mysql> use akademik;
Database changed
mysql> drop database akademik;
Query OK, 0 rows affected (0.02 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| test               |
+--------------------+
3 rows in set (0.00 sec)
```

5. See what tables are already active in database

   SHOW tables;

```
mysql> use akademik;
Database changed
mysql> show tables;
Empty set (0.00 sec)
```

6. Create Table

   CREATE TABLE <table_name> (<column_name><datatype>);

   CREATE TABLE mhs (NIM CHAR (8) NOT NULL PRIMARY KEY, nama_mhs VARCHAR (30), alamat VARCHAR (35));

   PRIMARY KEY is a primary key, in each table there must be at least one column that is used as PRIMARY KEY.

   NOT NULL means that each column must not be empty, whereas if allowed to be left blank can use parameters NULL.

7. Show the structure of the table

   DESCRIBE <nama_tabel>;

   DESC <nama_tabel>;

   DESCRIBE mhs;

   DESC mhs;

```
mysql> create table mhs (nim char(8)not null primary key, nama__mhs varchar(30),
 alamat varchar(35));
Query OK, 0 rows affected (0.08 sec)

mysql> desc;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near '' at
line 1
mysql> desc mh;
ERROR 1146 (42S02): Table 'akademik.mh' doesn't exist
mysql> desc mhs;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| nim       | char(8)     | NO   | PRI | NULL    |       |
| nama__mhs | varchar(30) | YES  |     | NULL    |       |
| alamat    | varchar(35) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> _
```

8. Edit Table Structure

There are times when we realize that the table structure that was created needs to be improved, e.g. in terms of adding columns, changing the width of columns, deleting columns, and etc. By using this ALTER command, we can avoid deficiencies or errors when we create tables.

ALTER TABEL table_name [revision specification]

Parameter [revision specification] is an option used to change the structure of a table i.e. CHANGE, ADD, DROP.

ALTER TABLE table_name what_to_change

a. Add new column

Parameter used is ADD.

ALTER TABLE table_name ADD new_column type(length) [FIRST | AFTER old_column]

FIRST means the new column will be added in the first order. AFTER means that we put the new column after the designated column.

For example, if we want to add a new column that is *no_telp* in the Student Table with a position after the column / address field with type INT :

ALTER TABLE mhs ADD no_telp INT AFTER alamat;

```
mysql> alter table mhs add no_telp int after alamat;
Query OK, 0 rows affected (0.19 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc mhs;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| nim       | char(8)     | NO   | PRI | NULL    |       |
| nama__mhs | varchar(30) | YES  |     | NULL    |       |
| alamat    | varchar(35) | YES  |     | NULL    |       |
| no_telp   | int(11)     | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
4 rows in set (0.02 sec)
```

b. Edit Column Name

Parameter used is CHANGE.

ALTER TABEL table_name CHANGE old_column new_column type (length)

e.g. ALTER TABLE mhs CHANGE alamat alamat_mhs VARCHAR (35)

```
mysql> alter table mhs change alamat alamat_mhs varchar(35);
Query OK, 0 rows affected (0.25 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc mhs;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| nim        | char(8)     | NO   | PRI | NULL    |       |
| nama__mhs  | varchar(30) | YES  |     | NULL    |       |
| alamat_mhs | varchar(35) | YES  |     | NULL    |       |
| no_telp    | int(11)     | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

c. Rename a Table Name

Parameter used is RENAME.

ALTER TABLE old_name RENAME [TO] new_name;

ALTER TABLE mhs RENAME TO mahasiswa;

```
mysql> alter table mhs rename to mahasiswa;
Query OK, 0 rows affected (0.06 sec)

mysql> show tables;
+--------------------+
| Tables_in_akademik |
+--------------------+
| mahasiswa          |
+--------------------+
1 row in set (0.00 sec)
```

d. Delete or eliminate components in the table

Removing this can include removing primary keys, columns, tables, indexes in columns.

To eliminate PRIMARY KEY:

ALTER TABLE nama_tabel DROP PRIMARY KEY;

To remove of one of the columns in the table:

ALTER TABLE mahasiswa DROP no_telp;

```
mysql> alter table mahasiswa drop no_telp;
Query OK, 0 rows affected (0.17 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc mahasiswa;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| nim        | char(8)     | NO   | PRI | NULL    |       |
| nama__mhs  | varchar(30) | YES  |     | NULL    |       |
| alamat_mhs | varchar(35) | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

5.4.2. DML Commands

1. Insert Data into Tabel

There are several ways to enter data: (1) by matching columns and data, (2) mentioning the columns, (3) without mentioning columns, and (4) entering only as columns. Below are the commands:

(1) By matching Columns and Data:

```
INSERT INTO nama_tabel SET
        First_column = 'data_first_column',
        Second_column = 'data_second_column',
        Last_column = 'data_last_column' ;
```

e.g.

INSERT INTO mhs SET NIM = ‘11010010’, nama_mhs =’M Rauf’, alamat_mhs=’Puri Kencana B2 Sleman’;

(2) By mentioning the columns:

INSERT INTO table_name (first_column, second_column, last_column) VALUES (data_first_column, data_second_column, data_last_column);

e.g.

INSERT INTO mhs (NIM, nama_mhs, alamat_mhs)
VALUES (‘11010010’, ’M Rauf’, ’Puri Kencana B2 Sleman’)

(3) Without Mentioning the Columns:

INSERT INTO table_name VALUES (data_first_column, data_second_column, data_last_column);

e.g.

INSERT INTO mhs VALUES (‘11010010’, ‘M Rauf’, ’Puri Kencana B2 Sleman’)

```
mysql> insert into mhs values (’11010010’, ’M Rauf’,’Puri Kencana B2 Sleman’);
Query OK, 1 row affected (0.03 sec)

mysql> insert into mhs values (’11010011’, ’Rafifah Azzahra’,’Blunyah Gede 227 J
ogja’);
Query OK, 1 row affected (0.33 sec)
```

2. Show Table Contents

This command is used to select or select or display data in a table. both display all columns, partial columns, and based on conditions.

a. Commands to display the whole data within a table:

SELECT * FROM <table_name>

SELECT * FROM mhs;

b. Commands to display the whole data by considering only selected column(s):

SELECT * from mhs WHERE nim = ‘11010010’

Show the student data named ‘M Rauf’:

SELECT * from mhs where nama_mhs = ‘M Rauf’

See data of students with address in Sleman:

SELECT * from mhs where alamat_mhs like ’%Sleman’

In the address criteria, you can see the use of the character ‘%’. This character implies, whatever the existing text will meet the criteria. So ‘% Sleman’ means all strings ended with word “Sleman”.

```
mysql> select * from mhs;
+----------+-----------------+-------------------------+
| nim      | nama_mhs        | alamat_mhs              |
+----------+-----------------+-------------------------+
| 11010010 | M Rauf          | Puri Kencana B2 Sleman  |
| 11010011 | Rafifah Azzahra | Blunyah Gede 227 Jogja  |
+----------+-----------------+-------------------------+
2 rows in set (0.02 sec)

mysql> select * from mhs where nim='11010010';
+----------+-----------+-------------------------+
| nim      | nama_mhs  | alamat_mhs              |
+----------+-----------+-------------------------+
| 11010010 | M Rauf    | Puri Kencana B2 Sleman  |
+----------+-----------+-------------------------+
1 row in set (0.00 sec)
```

c. Limit the number of fields read

To view specific fields from the table, replace the ‘*’ character with the desired field name. For example, showing only students and student names:

SELECT nim, nama_mhs from mhs;

3. Menampilkan data berurutan

The order by clause is used to sort the requested data by query. For example, you are asked to display the number and name of the student in order by number in ascending:

SELECT nim, nama_mhs from mhs order by nim asc;

If in descending:

SELECT nim, nama_mhs from mhs order by nim desc;

```
mysql> select nim, nama_mhs from mhs order by nim asc;
+----------+-----------------+
| nim      | nama_mhs        |
+----------+-----------------+
| 11010010 | M Rauf          |
| 11010011 | Rafifah Azzahra |
| 11010012 | M Tedy Farhan   |
| 11010013 | Shafa Dian      |
| 11010014 | Austin A Cetta  |
+----------+-----------------+
5 rows in set (0.00 sec)

mysql> select nim, nama_mhs from mhs order by nim desc;
+----------+-----------------+
| nim      | nama_mhs        |
+----------+-----------------+
| 11010014 | Austin A Cetta  |
| 11010013 | Shafa Dian      |
| 11010012 | M Tedy Farhan   |
| 11010011 | Rafifah Azzahra |
| 11010010 | M Rauf          |
+----------+-----------------+
5 rows in set (0.00 sec)
```

4. Change the contents of data in Table

   UPDATE table_name SET

   First_column = 'data_first_column',

   Second_column = 'data_second_column',

   Last_column = 'data_last_column',

   WHERE condition

   Using the UPDATE command without the WHERE clause causes all data in one column to be changed. For example, the name 'M Rauf' will be changed into 'Muhammad Rauf':

   UPDATE mhs SET

   nama_mhs = 'M Rauf'

   WHERE nim = '11010010'

```
mysql> update mhs set nama__mhs = 'Muhammad Rauf' where nim='11010010';
Query OK, 1 row affected (0.34 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select nim, nama__mhs from mhs order by nim desc;
+----------+-----------------+
| nim      | nama__mhs       |
+----------+-----------------+
| 11010014 | Austin A Cetta  |
| 11010013 | Shafa Dian      |
| 11010012 | M Tedy Farhan   |
| 11010011 | Rafifah Azzahra |
| 11010010 | Muhammad Rauf   |
+----------+-----------------+
5 rows in set (0.00 sec)
```

```
UPDATE   employees
SET      job_id   = (SELECT  job_id
                     FROM    employees
                     WHERE   employee_id = 205),
         salary   = (SELECT  salary
                     FROM    employees
                     WHERE   employee_id = 205)
WHERE    employee_id     =  114;
1 rows updated
```

5. Delete the contents of the table

   DELETE FROM table_name WHERE condition

```
 DELETE FROM departments
 WHERE  department_name = 'Finance';
1 rows deleted
```

5.5. EXERCISES

The assignment contains a post test that must be done by students as an evaluation of the practicum done (example evaluation sheet attached).

EXAMPLE OF
PRE-TEST / POST-TEST / EVALUATION ANSWER SHEET OF PRACTICUM 3: PRACTICUM NAME

Name :	**Pract Assistant:**	**Date:**
Student Number :	**Signature:**	**Score:**

MEETING 6: Aggregation Function

Meeting : 6

Time Allocation : 150 minutes

- Pre-Test : 30 minutes
- Practicum : 60 minutes
- Post-Test : 30 minutes

Scoring : 100%

- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

6.1. GOALS AND PERFORMANCE INDICATORS

By following this session, students are expected:

1. Able to implement aggregation functions with SQL commands

Performance indicators are measured by:

1. AVG, Count, Max, Min, Sum

6.2. STUDY LITERATURE

Aggregation functions include:

1. Count the number of records
2. Calculating the total value of an attribute
3. Calculate the average attribute value
4. Finding the greatest value of the attribute value
5. Finding the smallest value of the attribute value

Here's the formal query language for the above function:

KLAUSA	**PENJELASAN**
AVG	Sama dengan
COUNT	Mengetahui jumlah record
MAX	Mengetahui nilai maximal

MIN	Mengetahui nilai minimal
SUM	Menghitung jumlah data

Here are some of the operators that Clause usually follows WHERE :

OPERATOR	**PENJELASAN**
=	Sama dengan
< >, !=	Tidak sama dengan
<	Kurang dari
>	Lebih besar dari
< =	Kurang dari atau sama dengan
> =	Lebih dari atau sama dengan
! >	Tidak lebih besar dari
! <	Tidak lebih kecil dari
BETWEEN	Antara dua nilai yang ditentukan
LIKE	Menyesuaikan nilai yang ditentukan
IS NULL	Nilainya adalah NULL
IN	Nilainya ditentukan dalam sebuah daftar
NOT	Negasi dari sebuah operator perbandingan
AND	Merangkai kriteria pencarian
OR	Memastikan bahwa criteria pencarian adalah eksklusif

6.3. TOOLS

Material used in this chapter:

1. Computer
2. XAMPP
3. Web Browser

6.4. PRACTICUM INSTRUCTION

Make table mata_kuliah and insert table like picture in the below :

```
mysql> select * from mata_kuliah;
+----------+---------------+------+------+
| kode_kul | nama_kul      | sks  | sem  |
+----------+---------------+------+------+
| IT0101   | Logika Inf    | 3    |    1 |
| IT0102   | Studi Islam I | 2    |    1 |
| IT0103   | Kalkulus Inf  | 3    |    1 |
| IT0301   | Sertifikasi   | 0    |    3 |
| IT0401   | Basis Data    | 3    |    4 |
| IT0801   | Tugas Akhir   | 6    |    8 |
+----------+---------------+------+------+
6 rows in set (0.00 sec)
```

1. Displays data on courses held in semester 1.

>SELECT * FROM mata_kuliah WHERE sem = 1;

2. Displays data on courses undertaken other than semester 1.

>SELECT * FROM mata_kuliah WHERE sem <> 1;

3. Displaying course data containing informatics titles

>SELECT * FROM mat_kul WHERE nama_kul LIKE '% informatika%'

```
mysql> select * from mata_kuliah where sem=1;
+----------+---------------+------+------+
| kode_kul | nama_kul      | sks  | sem  |
+----------+---------------+------+------+
| IT0101   | Logika Inf    | 3    |    1 |
| IT0102   | Studi Islam I | 2    |    1 |
| IT0103   | Kalkulus Inf  | 3    |    1 |
+----------+---------------+------+------+
3 rows in set (0.00 sec)

mysql> select * from mata_kuliah where sem<>1;
+----------+-------------+------+------+
| kode_kul | nama_kul    | sks  | sem  |
+----------+-------------+------+------+
| IT0301   | Sertifikasi | 0    |    3 |
| IT0401   | Basis Data  | 3    |    4 |
| IT0801   | Tugas Akhir | 6    |    8 |
+----------+-------------+------+------+
3 rows in set (0.01 sec)

mysql> select * from mata_kuliah where nama_kul like '%Inf';
+----------+--------------+------+------+
| kode_kul | nama_kul     | sks  | sem  |
+----------+--------------+------+------+
| IT0101   | Logika Inf   | 3    |    1 |
| IT0103   | Kalkulus Inf | 3    |    1 |
+----------+--------------+------+------+
2 rows in set (0.00 sec)
```

1. Displays course data containing informatics titles and credits = 3

SELECT * FROM mat_kul WHERE nama_kul LIKE '% informatics%' AND sks = 3

2. Calculating the amount of data for the course

SELECT COUNT (*) from mat_kul;

3. Calculating the lowest credits, the most credits and the average credits

SELECT MIN (sks), MAX (sks), AVG (sks) from mat_kul;

4. Calculating the total number of credits

SELECT SUM (sks) from mat_kul;

```
mysql> select count(*) from mata_kuliah;
+----------+
| count(*) |
+----------+
|        6 |
+----------+
1 row in set (0.00 sec)

mysql> select min(sks), max(sks), avg(sks) from mata_kuliah;
+----------+----------+------------------+
| min(sks) | max(sks) | avg(sks)         |
+----------+----------+------------------+
| 0        | 6        | 2.8333333333333 |
+----------+----------+------------------+
1 row in set (0.00 sec)

mysql> select sum(sks) from mata_kuliah;
+----------+
| sum(sks) |
+----------+
|       17 |
+----------+
1 row in set (0.00 sec)
```

6.5. EXERCISES

The assignment contains a post test that must be done by students as an evaluation of the practicum done (example evaluation sheet attached).

EXAMPLE OF
PRE-TEST / POST-TEST / EVALUATION ANSWER SHEET OF PRACTICUM 3: PRACTICUM NAME

Name :	**Pract Assistant:**	**Date:**
Student Number :	**Signature:**	**Score:**

MEETING 7: Query of Table Relations

Meeting : 7

Time Allocation : 150 minutes

- Pre-Test : 30 minutes
- Practicum : 60 minutes
- Post-Test : 30 minutes

Scoring : 100%

- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

7.1. GOALS AND PERFORMANCE INDICATORS

By following this session, students are expected:

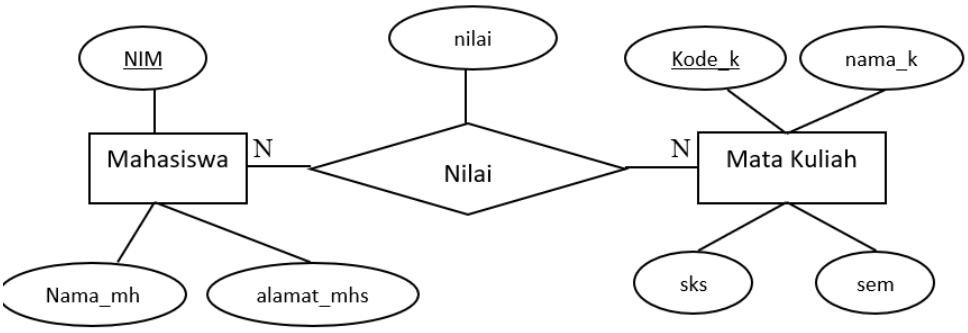
1. Able to implement aggregation functions with SQL commands

Performance indicators are measured by:

1. AVG, Count, Max, Min, Sum

7.2. STUDY LITERATURE

In designing a database with ERD will reflect the mapping table that will be formed on a physical database. As shown by the following ERD :



Mapping table from the ERD above is :

Mhs (nim, nama_mhs, alamat_mhs)

Mata_Kuliah (kode_kul, nama_kul, sks, sem)

Nilai (nim, kode_kul, nilai)

Querying 2 or more tables cannot be done carelessly. The tables that are the source of the query must have a relationship. NIM is the primary key of the student table and Kode_kul is the primary key of Mata_kuliah. The value table is a table that is the relation between the student table and mata_kuliah, so the primary key of the value table is taken from the primary key of the student table and mata_kuliah, namely nim and Kode_kul. This means that nim is a foreign key to the student table and also Kode_kul is a foreign key to the mata_kuliah table.

The command to display data from several tables, i.e. :

```
SELECT
 <tabel1>.<kolom_1>, <tabel1>.<kolom_2>, <tabel1>.<kolom_n>,
<tabel2>.<kolom_1>, <tabel2>.<kolom_2>, <tabel2>.<kolom_n>,
<tabeln>.<kolom_1>, <tabeln>.<kolom_2>, <tabeln>.<kolom_n>
FROM
<tabel1>, <tabel2>, <tabeln>
WHERE
<tabel1>.<kolom_x> = <tabel2>.<kolom_y> and
<tabel2>.<kolomy> = <tabeln>.<kolomn> and
<tabel1>.<kolomx> = <tabeln>.<kolomn>
```

The condition written in WHERE is an attribute that is a relation that connects between tabel 1, tabel 2 until tabel n.

For the example of the above table case, the WHERE command can be written as :

```
WHERE mhs.nim = nilai.nim and nilai.kode_kul = mata_kuliah.kode_kul
```

7.3. TOOLS

Material used in this chapter:

4. Computer
5. XAMPP
6. Web Browser

7.4. PRACTICUM INSTRUCTION

Create the following table :

Mhs (nim, nama_mhs, alamat_mhs)

Mata_Kuliah (kode_kul, nama_kul, sks, semester)

Nilai (nim, kode_kul, nilai)

```
mysql> show tables;
+--------------------+
| Tables_in_akademik |
+--------------------+
| mata_kuliah        |
| mhs                |
| nilai              |
+--------------------+
3 rows in set (0.00 sec)
```

Make a table of values by stating that nim and Kode_kul are foreign keys of the student table and mata_kuliah.

```
mysql> create table nilai (nim varchar(8), kode_kul varchar(6), nilai int, forei
gn key fk_nim(nim) references mhs(nim), foreign key fk_kode_kul(kode_kul) refere
nces mata_kuliah(kode_kul));
Query OK, 0 rows affected (0.09 sec)

mysql> desc nilai;
+----------+------------+------+-----+---------+-------+
| Field    | Type       | Null | Key | Default | Extra |
+----------+------------+------+-----+---------+-------+
| nim      | varchar(8) | YES  | MUL | NULL    |       |
| kode_kul | varchar(6) | YES  | MUL | NULL    |       |
| nilai    | int(11)    | YES  |     | NULL    |       |
+----------+------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> insert into nilai values ('11010010', 'IT0101',100);
Query OK, 1 row affected (0.03 sec)

mysql> insert into nilai values ('11010020', 'IT0101',100);
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint f
ails (`akademik/nilai`, CONSTRAINT `nilai_ibfk_1` FOREIGN KEY (`nim`) REFERENCES
 `mhs` (`nim`))
```

Fill in the table as shown below :

```
mysql> select * from mhs;
+----------+-----------------+-------------------------------+
| nim      | nama__mhs       | alamat_mhs                    |
+----------+-----------------+-------------------------------+
| 11010010 | M Rauf          | Puri Kencana B2 Sleman        |
| 11010011 | Rafifah Azzahra | Blunyah Gede 227 Sleman       |
| 11010012 | M Tedy Farhan   | Jl Boegenville IV No.9 Jambi  |
| 11010013 | Shafa Dian      | Jl Boegenville IV No.9 Jambi  |
| 11010014 | Austin A Cetta  | Palagan Regency B3 Jogja      |
+----------+-----------------+-------------------------------+
5 rows in set (0.00 sec)
```

```
mysql> select * from mata_kuliah;
+----------+---------------+------+------+
| kode_kul | nama_kul      | sks  | sem  |
+----------+---------------+------+------+
| IT0101   | Logika Inf    | 3    |    1 |
| IT0102   | Studi Islam I | 2    |    1 |
| IT0103   | Kalkulus Inf  | 3    |    1 |
| IT0301   | Sertifikasi   | 0    |    3 |
| IT0401   | Basis Data    | 3    |    4 |
| IT0801   | Tugas Akhir   | 6    |    8 |
+----------+---------------+------+------+
6 rows in set (0.00 sec)
```

```
mysql> select * from nilai;
+----------+----------+-------+
| nim      | kode_kul | nilai |
+----------+----------+-------+
| 11010010 | IT0101   |   100 |
| 11010011 | IT0101   |   100 |
| 11010010 | IT0401   |    90 |
| 11010011 | IT0401   |   100 |
| 11010012 | IT0401   |    80 |
| 11010013 | IT0401   |    85 |
+----------+----------+-------+
6 rows in set (0.00 sec)
```

Displays only student names and grades for courses with a code “IT0401”

```
mysql> select a.nama__mhs, b.nilai from mhs a, nilai b  where a.nim = b.nim and
b.kode_kul ='IT0401';
+-----------------+-------+
| nama__mhs       | nilai |
+-----------------+-------+
| M Rauf          |    90 |
| Rafifah Azzahra |   100 |
| M Tedy Farhan   |    80 |
| Shafa Dian      |    85 |
+-----------------+-------+
4 rows in set (0.00 sec)
```

Displays only the name of the course and its grades:

```
mysql> select a.nama_kul, b.nilai from mata_kuliah a, nilai b where a.kode_kul =
 b.kode_kul;
+------------+-------+
| nama_kul   | nilai |
+------------+-------+
| Logika Inf |   100 |
| Logika Inf |   100 |
| Basis Data |    90 |
| Basis Data |   100 |
| Basis Data |    80 |
| Basis Data |    85 |
+------------+-------+
6 rows in set (0.00 sec)
```

Displays the name of the course, student name and grades obtained:

```
mysql> select a.nama_kul, b.nilai, c.nama__mhs from mata_kuliah a, nilai b, mhs
c where a.kode_kul = b.kode_kul and b.nim = c.nim;
+------------+-------+-----------------+
| nama_kul   | nilai | nama__mhs       |
+------------+-------+-----------------+
| Logika Inf |   100 | M Rauf          |
| Logika Inf |   100 | Rafifah Azzahra |
| Basis Data |    90 | M Rauf          |
| Basis Data |   100 | Rafifah Azzahra |
| Basis Data |    80 | M Tedy Farhan   |
| Basis Data |    85 | Shafa Dian      |
+------------+-------+-----------------+
6 rows in set (0.00 sec)
```

Displays the name of the course, the name of the student and the values obtained which are sorted from the smallest value (the table column is adjusted as generated:

```
mysql> select a.nama_kul as 'Nama Mata Kuliah', b.nilai as 'Nilai Mata Kuliah',
c.nama__mhs as 'Nama Mahasiswa' from mata_kuliah a, nilai b, mhs c where a.kode_
kul = b.kode_kul and b.nim = c.nim order by nilai asc;
+------------------+-------------------+-----------------+
| Nama Mata Kuliah | Nilai Mata Kuliah | Nama Mahasiswa  |
+------------------+-------------------+-----------------+
| Basis Data       |                80 | M Tedy Farhan   |
| Basis Data       |                85 | Shafa Dian      |
| Basis Data       |                90 | M Rauf          |
| Logika Inf       |               100 | Rafifah Azzahra |
| Basis Data       |               100 | Rafifah Azzahra |
| Logika Inf       |               100 | M Rauf          |
+------------------+-------------------+-----------------+
6 rows in set (0.00 sec)
```

7.5. EXERCISES

The assignment contains a post test that must be done by students as an evaluation of the practicum done (example evaluation sheet attached).

EXAMPLE OF
PRE-TEST / POST-TEST / EVALUATION ANSWER SHEET OF PRACTICUM 3: PRACTICUM NAME

Name :	**Pract Assistant:**	**Date:**
Student Number :	**Signature:**	**Score:**

MEETING 8: Indexing

Meeting : 8

Time Allocation : 150 minutes

- Pre-Test : 30 minutes
- Practicum : 60 minutes
- Post-Test : 30 minutes

Scoring : 100%

- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

8.1. GOALS AND PERFORMANCE INDICATORS

By following this session, students are expected:

1. Able to Understanding index in database

Performance indicators are measured by:

1. Indexing in database

8.2. STUDY LITERATURE

As we already know in databases, tables are used to store data sets with the same type and schema. While Query is a method of retrieving data from one or more tables. We know SQL (Structured Query Language) as a structured "language" for performing Queries.

We have a table with a name ***mahasiswa***

NIM	Nama_Mahasiswa	Jurusan	Fakultas
061	Anto	Teknik Informatika	FTI
078	Hanifah	Teknik Informatika	FTI

when we run the SQL statement below :

```
Select * From mahasiswa Where Nama_Mahasiswa = 'Hanifah'
```

In simple terms, when the SQL statement above is executed, RDBMS will explore all the data in the *Mahasiswa* table, from first to last and perform filters (search data with Nama_Mahasiswa = "Hanifah") in each row. Then the rows of data that meet the requirements will be collected as a result.

With relatively small data, the query process looks very fast. But, with the simple method above, will the query process take the same time if the number of rows that need to be explored is multiplied? certainly not. Because the bigger the data, the bigger the query time. Then data that is not sequential will also be ineffective and As we know, that one of the biggest computational costs is disk I / O (see image below). The process of writing / reading a file on the hard disk takes a lot of time & resources.

a. Index

Have you ever used an English-Indonesian dictionary or something else, surely in his book there are alphabets a to z which are pasted on the side to make it easier for us to look for words. That's a simple explanation of how index works.

In a database, index is a data structure that contains a collection of keys and their references to the actual data in the table. The goal is to speed up the process of determining the location of the data without doing a full search of all data (full scan)

This index feature can solve existing problems such as the length of time to perform queries and so on

8.3. TOOLS

Material used in this chapter:

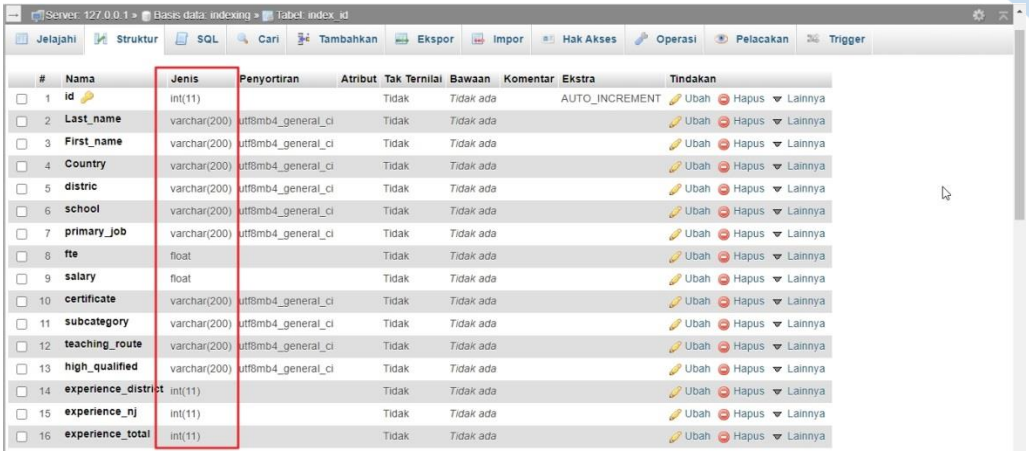
1. Computer
2. XAMPP
3. Web Browser

8.4. PRACTICUM INSTRUCTION

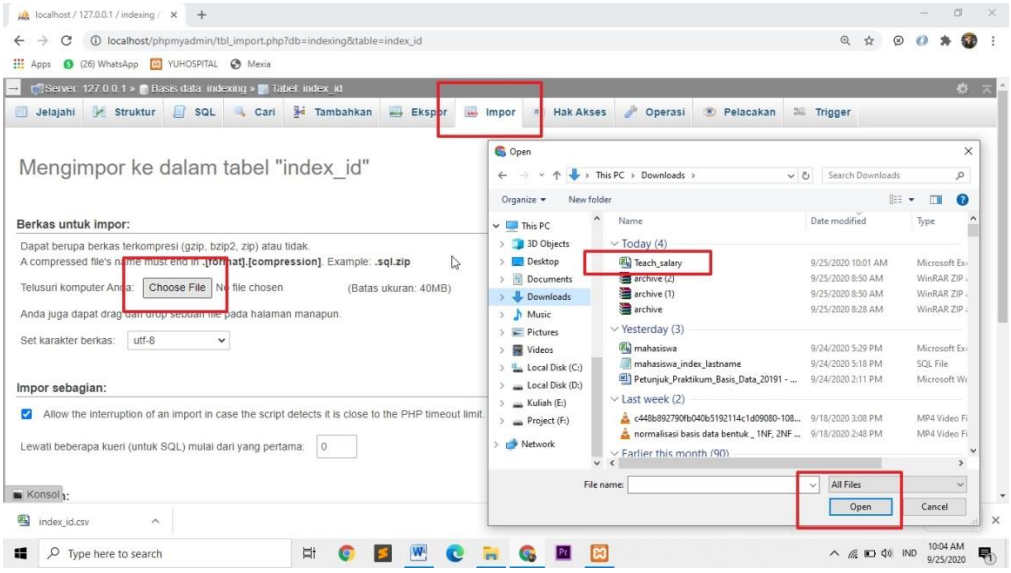
8.4.1. Indexing

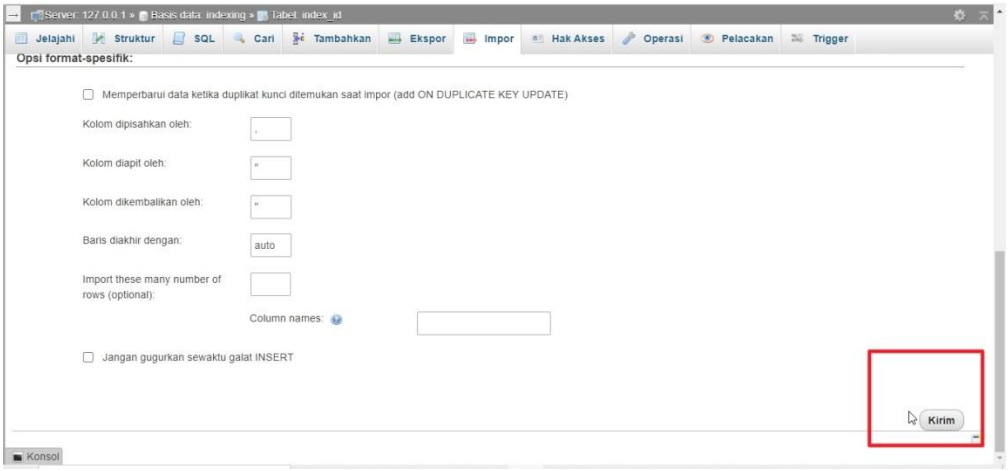
1. Open phpmy admin,create database give name ***indexing***

2. Create table give name ***index_id***,create table with structure same like this,

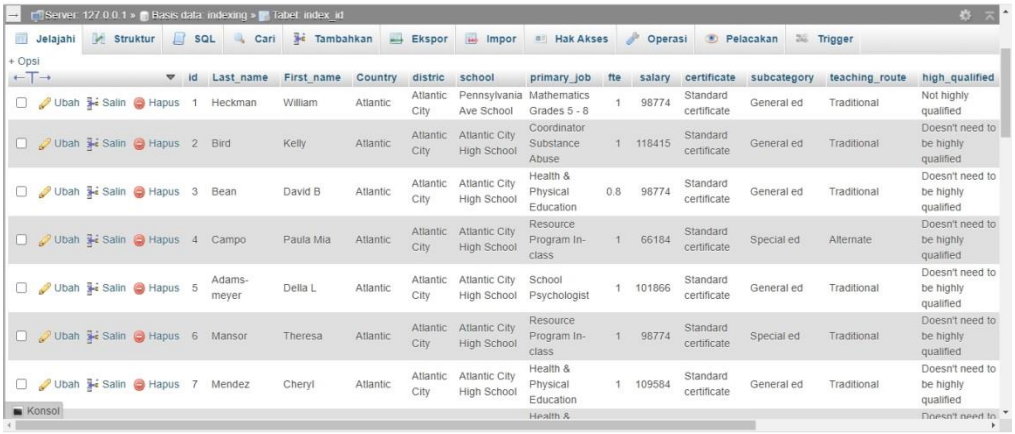


3. And then download dataset in **bit.ly/Database_Meet8,** download file dataset Teach_salary.csv
4. After download the file, then export dataset to table ***index_id*** like this :

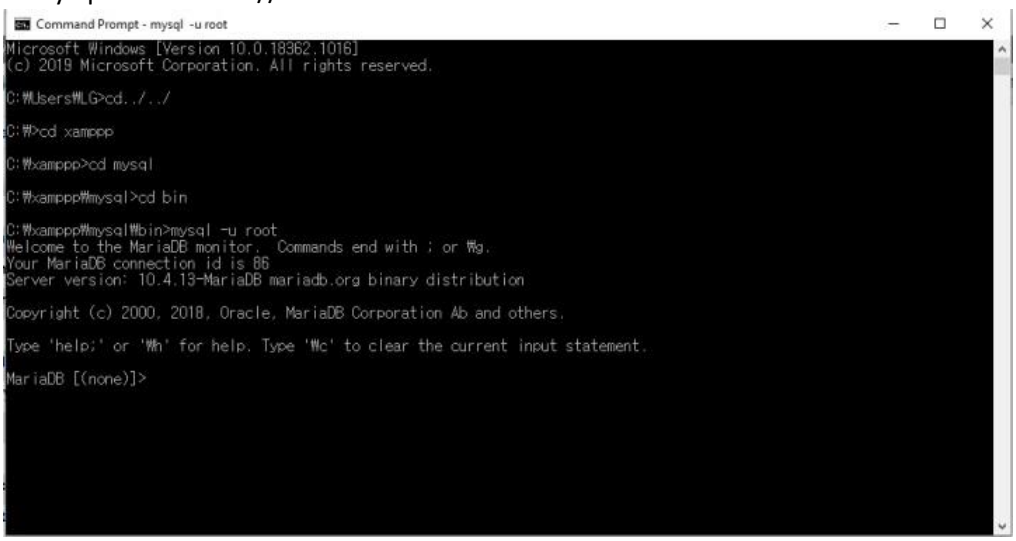




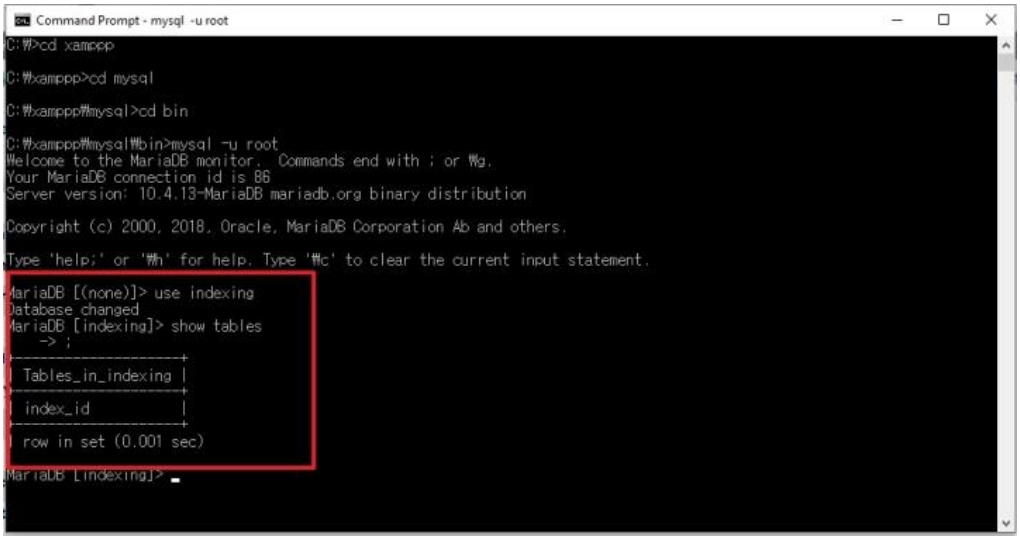
if the export is successful, the data will appear as follows



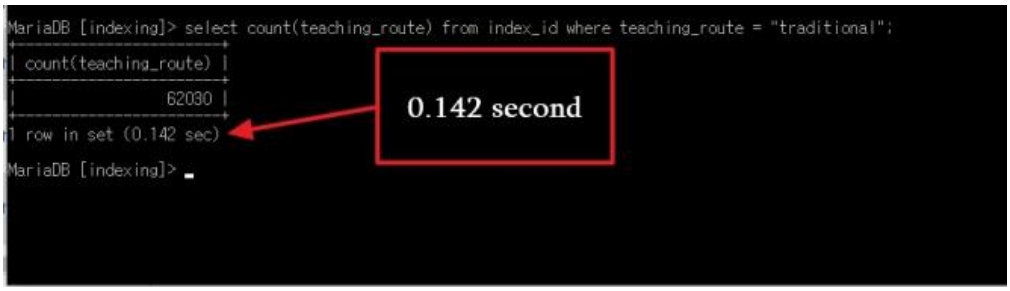
5. Open cmd and go to mysql.exe
   In my computer the directory like this C:/xammp/mysql/bin/mysql.exe
   User = “root”
   Password = “ ”
   >mysql.exe –u root //click enter



Use indexing database, in indexing database containt index_id table

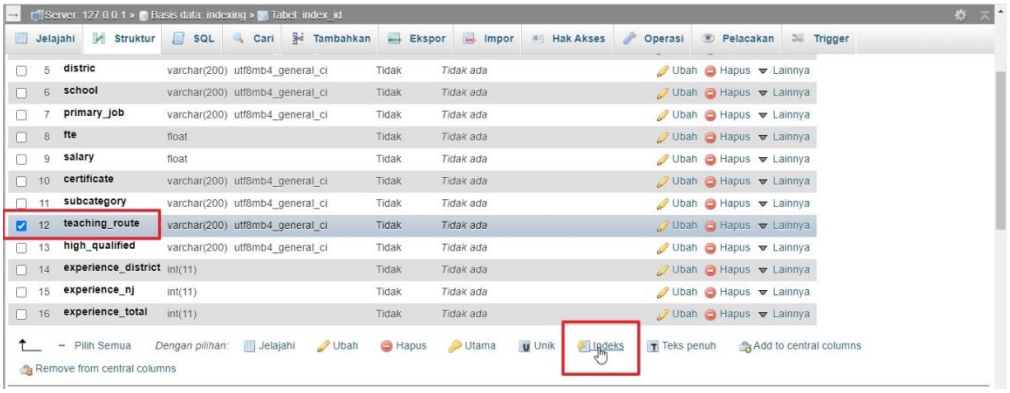


6. You can try using this stetement
   >select count(teaching_route) from index_id where teaching_route =”traditional”;
   //This is a statement that functions to calculate the amount of traditional teaching in the index_id table **without index**

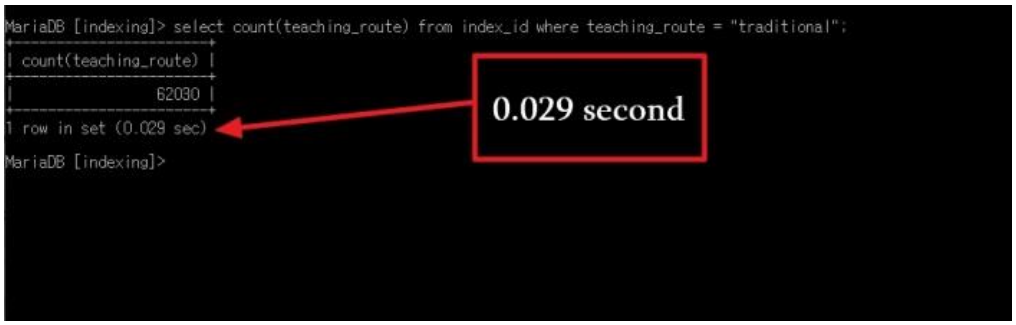


the time to calculate it is 0.142 second,

7. now we will calculate it but using index, open phpmyadmin again,klik teaching_route and then klick index.
   //You also can use this statement ini cmd
   >ALTER TABLE `index_id` ADD INDEX(`teaching_route`);



8. try to do the recalculation and see the time used, the time used is shorter
   this may not be the difference much, but if the data is large and complex, then this will be very useful



8.5. EXERCISES

The assignment contains a post test that must be done by students as an evaluation of the practicum done (example evaluation sheet attached).

EXAMPLE OF
PRE-TEST / POST-TEST / EVALUATION ANSWER SHEET OF PRACTICUM 3: PRACTICUM NAME

Name :	**Pract Assistant:**	**Date:**
Student Number :	**Signature:**	**Score:**

MEETING 9: Table Relations with Join

Meeting : 9

Time Allocation : 150 minutes

- Pre-Test : 30 minutes
- Practicum : 60 minutes
- Post-Test : 30 minutes

Scoring : 100%

- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

9.1. GOALS AND PERFORMANCE INDICATORS

By following this session, students are expected:

9. Able to do data searches between tables

Performance indicators are measured by:

1. join, left join, right join, full join, union

9.2. STUDY LITERATURE

The join mechanism is used to find data from several tables based on the logical relationship of these tables. Types of Join, are:

1. Inner Join is a set in which is the combined result of two interconnected tables for all records in pairs
2. Full Outer Join returns all rows from both tables
3. Left Outer Join produces all the table rows to the left of the statement, and the corresponding rows from the table to the right of the statement
4. Right Outer Join produces all the table rows to the left of the statement, and the corresponding rows from the table to the left of the statement
5. Union is used to combine two query operations into one cursor

9.3. TOOLS

Material used in this chapter:

1. Computer
2. XAMPP
3. Web Browser

9.4. PRACTICUM INSTRUCTION

```
mysql> select * from mhs;
+----------+-----------------+-------------------------------+
| nim      | nama_mhs        | alamat_mhs                    |
+----------+-----------------+-------------------------------+
| 11010010 | Muhammad Rauf   | Puri Kencana B2 Sleman        |
| 11010011 | Rafifah Azzahra | Blunyah Gede 227 Sleman       |
| 11010012 | M Tedy Farhan   | Jl Boegenville IV No.9 Jambi  |
| 11010013 | Shafa Dian      | Jl Boegenville IV No.9 Jambi  |
| 11010014 | Austin A Getta  | Palagan Regency B3 Jogja      |
| 11010015 | Luna Ramadhan   | Jl Keparakan No.1 Samarinda   |
+----------+-----------------+-------------------------------+
6 rows in set (0.00 sec)
```

```
mysql> select * from nilai;
+----------+----------+-------+
| nim      | kode_kul | nilai |
+----------+----------+-------+
| 11010010 | IT0101   |   100 |
| 11010011 | IT0101   |   100 |
| 11010010 | IT0401   |    90 |
| 11010011 | IT0401   |   100 |
| 11010012 | IT0401   |    80 |
| 11010013 | IT0401   |    85 |
+----------+----------+-------+
6 rows in set (0.00 sec)
```

```
mysql> select * from nilai;
+----------+----------+-------+
| nim      | kode_kul | nilai |
+----------+----------+-------+
| 11010010 | IT0101   |   100 |
| 11010011 | IT0101   |   100 |
| 11010010 | IT0401   |    90 |
| 11010011 | IT0401   |   100 |
| 11010012 | IT0401   |    80 |
| 11010013 | IT0401   |    85 |
+----------+----------+-------+
6 rows in set (0.00 sec)
```

1. Inner Join

   Displays the name of the student, the course code follows the grades obtained

   With the command relation attribute between tables

```
mysql> select mhs.nim, mhs.nama_mhs, nilai.kode_kul, nilai.nilai
    -> from mhs, nilai
    -> where mhs.nim = nilai.nim;
+----------+-----------------+----------+-------+
| nim      | nama_mhs        | kode_kul | nilai |
+----------+-----------------+----------+-------+
| 11010010 | Muhammad Rauf   | IT0101   |   100 |
| 11010010 | Muhammad Rauf   | IT0401   |    90 |
| 11010011 | Rafifah Azzahra | IT0101   |   100 |
| 11010011 | Rafifah Azzahra | IT0401   |   100 |
| 11010012 | M Tedy Farhan   | IT0401   |    80 |
| 11010013 | Shafa Dian      | IT0401   |    85 |
+----------+-----------------+----------+-------+
6 rows in set (0.00 sec)
```

With the inner join command

```
mysql> select mhs.nim, mhs.nama__mhs, nilai.kode_kul, nilai.nilai
    -> from mhs inner join nilai
    -> on mhs.nim = nilai.nim;
+----------+-----------------+----------+-------+
| nim      | nama__mhs       | kode_kul | nilai |
+----------+-----------------+----------+-------+
| 11010010 | Muhammad Rauf   | IT0101   |   100 |
| 11010010 | Muhammad Rauf   | IT0401   |    90 |
| 11010011 | Rafifah Azzahra | IT0101   |   100 |
| 11010011 | Rafifah Azzahra | IT0401   |   100 |
| 11010012 | M Tedy Farhan   | IT0401   |    80 |
| 11010013 | Shafa Dian      | IT0401   |    85 |
+----------+-----------------+----------+-------+
6 rows in set (0.00 sec)
```

The two pictures above produce the same results. It can be seen that in the student table and mata_kuliah, nim is an attribute that connects to the two tables. In the command above only paired records are raised. In the student table there are records with nim '11010014' and '11010015' which are not in the value table, so they are not displayed.

2. Left Join

Displaying all student data in the student table and student grades data.

```
mysql> select mhs.nim, mhs.nama__mhs, nilai.kode_kul, nilai.nilai from mhs left
join nilai on mhs.nim = nilai.nim;
+----------+-----------------+----------+-------+
| nim      | nama__mhs       | kode_kul | nilai |
+----------+-----------------+----------+-------+
| 11010010 | Muhammad Rauf   | IT0101   |   100 |
| 11010010 | Muhammad Rauf   | IT0401   |    90 |
| 11010011 | Rafifah Azzahra | IT0101   |   100 |
| 11010011 | Rafifah Azzahra | IT0401   |   100 |
| 11010012 | M Tedy Farhan   | IT0401   |    80 |
| 11010013 | Shafa Dian      | IT0401   |    85 |
| 11010014 | Austin A Cetta  | NULL     |  NULL |
| 11010015 | Luna Ramadhan   | NULL     |  NULL |
+----------+-----------------+----------+-------+
8 rows in set (0.00 sec)
```

It will be seen that the data table to the right will be filled with NULL because the student in question has no value, while the data from the left table will be displayed in full.

3. Right Join

Displays all data values in the following value table student name.

```
mysql> select mhs.nim, mhs.nama__mhs, nilai.kode_kul, nilai.nilai from mhs right
 join nilai on mhs.nim = nilai.nim;
+----------+-----------------+----------+-------+
| nim      | nama__mhs       | kode_kul | nilai |
+----------+-----------------+----------+-------+
| 11010010 | Muhammad Rauf   | IT0101   |   100 |
| 11010011 | Rafifah Azzahra | IT0101   |   100 |
| 11010010 | Muhammad Rauf   | IT0401   |    90 |
| 11010011 | Rafifah Azzahra | IT0401   |   100 |
| 11010012 | M Tedy Farhan   | IT0401   |    80 |
| 11010013 | Shafa Dian      | IT0401   |    85 |
+----------+-----------------+----------+-------+
6 rows in set (0.00 sec)
```

It will be seen that the data from the right hand table ie the value table will be displayed in its entirety.

4. Union

```
mysql> select * from mhs union select * from nilai;
+----------+-----------------+-------------------------------+
| nim      | nama_mhs        | alamat_mhs                    |
+----------+-----------------+-------------------------------+
| 11010010 | Muhammad Rauf   | Puri Kencana B2 Sleman        |
| 11010011 | Rafifah Azzahra | Blunyah Gede 227 Sleman       |
| 11010012 | M Tedy Farhan   | Jl Boegenville IV No.9 Jambi  |
| 11010013 | Shafa Dian      | Jl Boegenville IV No.9 Jambi  |
| 11010014 | Austin A Cetta  | Palagan Regency B3 Jogja      |
| 11010015 | Luna Ramadhan   | Jl Keparakan No.1 Samarinda   |
| 11010010 | IT0101          | 100                           |
| 11010011 | IT0101          | 100                           |
| 11010010 | IT0401          | 90                            |
| 11010011 | IT0401          | 100                           |
| 11010012 | IT0401          | 80                            |
| 11010013 | IT0401          | 85                            |
+----------+-----------------+-------------------------------+
12 rows in set (0.02 sec)
```

9.5. EXERCISES

The assignment contains a post test that must be done by students as an evaluation of the practicum done (example evaluation sheet attached).

EXAMPLE OF
PRE-TEST / POST-TEST / EVALUATION ANSWER SHEET OF PRACTICUM 3: PRACTICUM NAME

Name :	**Pract Assistant:**	**Date:**
Student Number :	**Signature:**	**Score:**

MEETING 10: Subquery

Meeting : 10

Time Allocation : 150 minutes

- Pre-Test : 30 minutes
- Practicum : 60 minutes
- Post-Test : 30 minutes

Scoring : 100%

- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

10.1. GOALS AND PERFORMANCE INDICATORS

By following this session, students are expected:

1. Able to understand the concept of SubQuery
2. Able to create a SubQuery command

Performance indicators are measured by:

1. IN, NOT IN, EXIST, Operator Perbandingan

10.2. STUDY LITERATURE

Sub Query is a query that is part of a query, used to handle complex problems that may be difficult to do with just a query. Sub Queries provide an alternative way to perform operations that require complex joins or unions. Some rules:

1. In a query, it can have more than 1 sub query.
2. A sub query may have more sub queries.
3. The comparison operators that can be used are =,>, <,> =, <=, <>,! =, <=>, IN, ANY, SOME, ALL, EXISTS, NOT EXISTS.

For operator =,>, <,> =, <=, <>, ! =, <=> the sub query can only have 1 line, if the line has more than 1 line, it will display the message "Subquery returns more than 1 row".

The IN operator checks whether a value in the outer query is in a sub query result. Sub queries may have more than 1 row of data. The IN operator can be compared to the "= ANY" operator. The opposite result of an IN operation is NOT IN. NOT IN operator can be equated with "<> ALL". The EXISTS operator is used to check whether a subquery has a row or not. If there is at least 1 line (even though it only contains NULL), it will be TRUE. NOT EXISTS is the opposite of EXISTS.

10.3. TOOLS

Material used in this chapter:

1. Computer
2. XAMPP
3. Web Browser

10.4. PRACTICUM INSTRUCTION

Table of Mhs (NIM, Nama_mhs, Alamat_mhs)

Table of Nilai (NIM, Nilai, Kode_kul)

Table of Kuliah (NIM, Kode_kul)

Fill in the value table as below :

```
mysql> select * from nilai;
+----------+----------+-------+
| nim      | kode_kul | nilai |
+----------+----------+-------+
| 11010010 | IT0101   |    80 |
| 11010011 | IT0101   |    75 |
| 11010010 | IT0401   |    70 |
| 11010011 | IT0401   |    75 |
| 11010012 | IT0401   |    70 |
| 11010013 | IT0401   |    70 |
+----------+----------+-------+
6 rows in set (0.00 sec)
```

1. Display data on student grades that exceed the average grade of the whole course

   Queries require Sub Queries because in order to find the desired data, the average course must be sought first. Sub query works to find the average value of the course that will be a benchmark in the selection of value data

   Sub Query : SELECT avg(nilai) from nilai

```
mysql> select avg(nilai) from nilai;
+-------------+
| avg(nilai) |
+-------------+
|    73.3333 |
+-------------+
1 row in set (0.05 sec)
```

   Main Query : SELECT nim, nilai FROM nilai WHERE nilai > (SELECT avg(nilai) from nilai)

```
mysql> select nim, nilai from nilai where nilai>(select avg(nilai) from nilai);
+----------+-------+
| nim      | nilai |
+----------+-------+
| 11010010 |    80 |
| 11010011 |    75 |
| 11010011 |    75 |
+----------+-------+
3 rows in set (0.03 sec)
```

2. Look for data values whose values are equal to the largest.

Queries require Sub Queries to be executed first because in order to find the desired value data, the largest value must be searched first. This query can be done using ORDER BY and LIMIT, but it will only produce 1 line. What if the data that fits the criteria is more than 1 row.

```
mysql> select * from nilai;
+----------+----------+-------+
| nim      | kode_kul | nilai |
+----------+----------+-------+
| 11010010 | IT0101   |    80 |
| 11010011 | IT0101   |    75 |
| 11010010 | IT0401   |    70 |
| 11010011 | IT0401   |    75 |
| 11010012 | IT0401   |    70 |
| 11010013 | IT0401   |    80 |
+----------+----------+-------+
6 rows in set (0.00 sec)
```

SELECT nim, nilai, kode_kul

FROM nilai WHERE nilai=(SELECT MAX(nilai)
FROM nilai)

```
mysql> select nim, nilai, kode_kul from nilai where nilai = (select max(nilai) f
rom nilai);
+----------+-------+----------+
| nim      | nilai | kode_kul |
+----------+-------+----------+
| 11010010 |    80 | IT0101   |
| 11010013 |    80 | IT0401   |
+----------+-------+----------+
2 rows in set (0.00 sec)
```

SELECT nim, nilai, kode_kul

FROM nilai ORDER BY nilai asc LIMIT 1

The first query might display data values of more than 1 row when rows whose values are equal to MAX values (values) of more than 1 row.

The second query will only display 1 line because there is a use of LIMIT. The disadvantage of this SQL is when there are data values that have the same value with MAX (value) of more than 1 row.

```
mysql> select * from nilai order by nilai desc limit 1;
+----------+----------+-------+
| nim      | kode_kul | nilai |
+----------+----------+-------+
| 11010010 | IT0101   |    80 |
+----------+----------+-------+
1 row in set (0.00 sec)
```

Displays data on students taking courses 'IT0401'

The query requires sub-queries because they must compare student data with student data in the lecture table. Means student data in the lecture table must be sought first.

SELECT nim, nama_mhs

FROM mhs

WHERE nim IN (SELECT nim FROM kuliah
where kode_kul = 'IT100')

Sub query works to look for student data that has been registered in the lecture table (taking the course)

```
mysql> select nim, nama__mhs from mhs where nim in (select nim from nilai where
kode_kul = 'IT0401');
+----------+-----------------+
| nim      | nama__mhs       |
+----------+-----------------+
| 11010010 | Muhammad Rauf   |
| 11010011 | Rafifah Azzahra |
| 11010012 | M Tedy Farhan   |
| 11010013 | Shafa Dian      |
+----------+-----------------+
4 rows in set (0.00 sec)
```

Displays data on students who did not take database courses:

```
mysql> select nim, nama__mhs from mhs where nim not in (select nim from nilai wh
ere kode_kul = 'IT0401');
+----------+----------------+
| nim      | nama__mhs      |
+----------+----------------+
| 11010014 | Austin A Cetta |
| 11010015 | Luna Ramadhan  |
+----------+----------------+
2 rows in set (0.00 sec)
```

10.5. EXERCISES

The assignment contains a post test that must be done by students as an evaluation of the practicum done (example evaluation sheet attached).

EXAMPLE OF
PRE-TEST / POST-TEST / EVALUATION ANSWER SHEET OF PRACTICUM 3: PRACTICUM NAME

Name :	**Pract Assistant:**	**Date:**
Student Number :	**Signature:**	**Score:**

REFERENCES

Silberschatz, A., Korth, H. F., & Sudarshan, S. (2011). Database System Concepts - 6th. ed. In *Database*. https://doi.org/10.1145/253671.253760


LABORATORIUM
TEKNIK INFORMATIKA
UNIVERSITAS AHMAD DAHLAN


UNIVERSITAS AHMAD DAHLAN